

Chapman & Hall/CRC
Numerical Analysis and Scientific Computing

CLASSICAL AND MODERN NUMERICAL ANALYSIS

THEORY, METHODS AND PRACTICE

Azmy S. Ackleh
Edward James Allen
Ralph Baker Hearfott
Padmanabhan Seshaiyer



CRC Press
Taylor & Francis Group

A CHAPMAN & HALL BOOK

CLASSICAL AND MODERN NUMERICAL ANALYSIS

THEORY, METHODS and PRACTICE

CHAPMAN & HALL/CRC

Numerical Analysis and Scientific Computing

Aims and scope:

Scientific computing and numerical analysis provide invaluable tools for the sciences and engineering. This series aims to capture new developments and summarize state-of-the-art methods over the whole spectrum of these fields. It will include a broad range of textbooks, monographs, and handbooks. Volumes in theory, including discretisation techniques, numerical algorithms, multiscale techniques, parallel and distributed algorithms, as well as applications of these methods in multi-disciplinary fields, are welcome. The inclusion of concrete real-world examples is highly encouraged. This series is meant to appeal to students and researchers in mathematics, engineering, and computational science.

Editors

Choi-Hong Lai
*School of Computing and
Mathematical Sciences
University of Greenwich*

Frédéric Magoulès
*Applied Mathematics and
Systems Laboratory
Ecole Centrale Paris*

Editorial Advisory Board

Mark Ainsworth
*Mathematics Department
Strathclyde University*

Peter Jimack
*School of Computing
University of Leeds*

Todd Arbogast
*Institute for Computational
Engineering and Sciences
The University of Texas at Austin*

Takashi Kako
*Department of Computer Science
The University of Electro-Communications*

Craig C. Douglas
*Computer Science Department
University of Kentucky*

Peter Monk
*Department of Mathematical Sciences
University of Delaware*

Ivan Graham
*Department of Mathematical Sciences
University of Bath*

Francois-Xavier Roux
ONERA

Arthur E.P. Veldman
*Institute of Mathematics and Computing Science
University of Groningen*

Proposals for the series should be submitted to one of the series editors above or directly to:

CRC Press, Taylor & Francis Group
4th, Floor, Albert House
1-4 Singer Street
London EC2A 4BQ
UK

Published Titles

Classical and Modern Numerical Analysis: Theory, Methods and Practice

*Azmy S. Ackleh, Edward James Allen, Ralph Baker Kearfott,
and Padmanabhan Seshaiyer*

A Concise Introduction to Image Processing using C++

Meiqing Wang and Choi-Hong Lai

Decomposition Methods for Differential Equations: Theory and Applications

Juergen Geiser

Grid Resource Management: Toward Virtual and Services Compliant Grid Computing

Frédéric Magoulès, Thi-Mai-Huong Nguyen, and Lei Yu

Introduction to Grid Computing

Frédéric Magoulès, Jie Pan, Kiat-An Tan, and Abhinit Kumar

Mathematical Objects in C++: Computational Tools in a Unified Object- Oriented Approach

Yair Shapira

Numerical Linear Approximation in C

Nabih N. Abdelmalek and William A. Malek

Numerical Techniques for Direct and Large-Eddy Simulations

Xi Jiang and Choi-Hong Lai

Parallel Algorithms

Henri Casanova, Arnaud Legrand, and Yves Robert

Parallel Iterative Algorithms: From Sequential to Grid Computing

Jacques M. Bahi, Sylvain Contassot-Vivier, and Raphael Couturier

CLASSICAL AND MODERN NUMERICAL ANALYSIS

THEORY, METHODS and PRACTICE

Azmy S. Ackleh

University of Louisiana
LaFayette, Louisiana, U.S.A.

Edward James Allen

Texas Tech Univeristy
Lubbock, Texas, U.S.A.

Ralph Baker Kearfott

University of Louisiana
LaFayette, Louisiana, U.S.A.

Padmanabhan Seshaiyer

George Mason University
Fairfax, Virginia, U.S.A.



CRC Press

Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A CHAPMAN & HALL BOOK

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

© 2009 by Taylor & Francis Group, LLC
CRC Press is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works
Version Date: 20131121

International Standard Book Number-13: 978-1-4200-9158-8 (eBook - PDF)

This book contains information obtained from authentic and highly regarded sources. Reasonable efforts have been made to publish reliable data and information, but the author and publisher cannot assume responsibility for the validity of all materials or the consequences of their use. The authors and publishers have attempted to trace the copyright holders of all material reproduced in this publication and apologize to copyright holders if permission to publish in this form has not been obtained. If any copyright material has not been acknowledged please write and let us know so we may rectify in any future reprint.

Except as permitted under U.S. Copyright Law, no part of this book may be reprinted, reproduced, transmitted, or utilized in any form by any electronic, mechanical, or other means, now known or hereafter invented, including photocopying, microfilming, and recording, or in any information storage or retrieval system, without written permission from the publishers.

For permission to photocopy or use material electronically from this work, please access www.copyright.com (<http://www.copyright.com/>) or contact the Copyright Clearance Center, Inc. (CCC), 222 Rosewood Drive, Danvers, MA 01923, 978-750-8400. CCC is a not-for-profit organization that provides licenses and registration for a variety of users. For organizations that have been granted a photocopy license by the CCC, a separate system of payment has been arranged.

Trademark Notice: Product or corporate names may be trademarks or registered trademarks, and are used only for identification and explanation without intent to infringe.

Visit the Taylor & Francis Web site at
<http://www.taylorandfrancis.com>

and the CRC Press Web site at
<http://www.crcpress.com>

Dedication

*To my wife Howayda,
my children Aseal and Wedad,
and my parents Sima'an and Wedad—A.S.A.*

*To my wife Linda
and my daughter Anna—E.J.A.*

*To my wife Ruth,
my daughter Frances,
and my mother Edith—R.B.K.*

*To my wife Revathy,
my daughters Pradyuta and Prasadha,
and my parents Usha and Seshaiyer—P.S.*

Preface

The purpose of this book is to provide a sound introduction to the theory, methods, and application of numerical analysis/computational mathematics. It is written with two primary objectives:

- (a) *to provide an advanced graduate introduction to the theory and the methods in numerical analysis that will help prepare students for taking doctoral examinations in numerical analysis; and*
- (b) *to provide a solid foundation in numerical analysis for more specialized topics such as finite-element theory and application, advanced numerical linear algebra, optimization, or approximation of stochastic differential equations.*

Indeed, this book provides useful background knowledge for graduate study in any area of applied mathematics.

The main topics in introductory numerical analysis include solution of non-linear equations, numerical linear algebra, ordinary differential equations, approximation theory, as well as, for example, numerical integration and boundary-value problems. These topics are introduced and examined in separate chapters. Many examples are described to illustrate the concepts. The emphasis in the explanations is to provide a good understanding of the concepts. At the end of each chapter, analytical and computational exercises are provided. The exercises are designed to complement the material in the text and to illustrate how the theory and methods can be applied. An interesting feature of this book is the presentation of interval computation in numerical analysis. Throughout the text, explanations of interval arithmetic, interval computation, and interval algorithms are provided.

There are many excellent books available on the theory, application, and computational methods of numerical analysis. Most of these texts, however, are either undergraduate texts with elementary theory and problems or specialized treatments, for example, on numerical linear algebra or differential equations. The bibliography lists many of these books. It is hoped that the present book will complement these previous books in providing a more advanced graduate-level introduction to the theory and methods in numerical analysis. Nevertheless, as numerical analysis is a large and rapidly expanding area of mathematics, we were forced to choose topics, computational methods, and analytical techniques that we thought would endure as well as provide a good background for understanding numerical techniques for more advanced problems.

One of the objectives of this book is to provide a clear and solid introduction to the theory and application of computational methods for applied mathematicians. The intent of this book is to provide a background to numerical methods so the reader will be in a position to apply the techniques and to understand the mathematical literature in this area, including more specialized texts. To understand the material presented in this book, proficiency in linear algebra and intermediate analysis is assumed. In particular, prerequisite courses for thoroughly understanding the concepts in this book include linear algebra, differential equations, advanced calculus, and intermediate analysis. In addition, some knowledge of scientific computing and programming is very helpful. Throughout the book, computational procedures are described, and to thoroughly understand these procedures, familiarity with some computer language such as MATLAB or Fortran is essential.

For those readers with access to MATLAB, we have provided a set of MATLAB functions and scripts implementing various computations described in the text. These are available, either individually or as a complete “zip” file, on the web page

<http://interval.louisiana.edu/Classical-and-Modern-NA/>

This web page gives short descriptions, as well as section numbers and exercise numbers corresponding to each function provided. It also cross-references section numbers to selected intrinsic capabilities of MATLAB, and does the same for interval arithmetic routines, publicly available for the book *Introduction to Interval Analysis*, by Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud, SIAM, Philadelphia, 2009.

We are grateful to the University of Louisiana at Lafayette, Texas Tech University, and to George Mason University for providing us with the opportunities to use this book in teaching both two-semester and one-semester (with selected topics) graduate courses in numerical analysis. We wish to thank our colleagues Christo Christov and Hongtao Yang for using this book in teaching the graduate numerical analysis course at UL Lafayette and for providing us with corrections. We are also grateful to the colleagues and graduate students who provided us with comments and corrections on the manuscript. In particular, we wish to thank Youssef Dib for his diligent work completing the figures, and Shuhua Hu and Xubo Wang for their help in typing some of these notes in $\text{\LaTeX}$. We also wish to thank Anthony Holmes and Mark Thompson, who went out of their way to supply many valuable corrections and suggestions. The students in our graduate-level numerical analysis course who, while learning the material from preliminary copies of the manuscript, also made numerous corrections and suggestions, are worthy of mention, specifically, Ashley Avery, Ross Chiquet, Mark Delcambre, Frank Hammers, Xiaodong Lian, Tchavdar Marinov, Julie Roy, Christine Terranova, and Xing Yang.

Contents

List of Figures	xvii
-----------------	------

List of Tables	xix
----------------	-----

1 Mathematical Review and Computer Arithmetic	1
1.1 Mathematical Review	1
1.1.1 Intermediate Value Theorem, Mean Value Theorems, and Taylor's Theorem	1
1.1.2 Big “ $\mathcal{O}$ ” and Little “ o ” Notation	5
1.1.3 Convergence Rates	7
1.2 Computer Arithmetic	8
1.2.1 Floating Point Arithmetic and Rounding Error	10
1.2.2 Practicalities and the IEEE Floating Point Standard .	17
1.3 Interval Computations	23
1.3.1 Interval Arithmetic	23
1.3.2 Application of Interval Arithmetic: Examples	28
1.4 Exercises	29
2 Numerical Solution of Nonlinear Equations of One Variable	35
2.1 Introduction	35
2.2 Bisection Method	35
2.3 The Fixed Point Method	39
2.4 Newton's Method (Newton-Raphson Method)	49
2.5 The Univariate Interval Newton Method	54
2.5.1 Existence and Uniqueness Verification	54
2.5.2 Interval Newton Algorithm	55
2.5.3 Convergence Rate of the Interval Newton Method . .	57
2.6 Secant Method and Müller's Method	58
2.6.1 The Secant Method	59
2.6.2 Müller's Method	63
2.7 Aitken Acceleration and Steffensen's Method	64
2.7.1 Aitken Acceleration	65
2.7.2 Steffensen's Method	68
2.8 Roots of Polynomials	69
2.9 Additional Notes and Summary of Chapter 2	74
2.10 Exercises	75

3	Numerical Linear Algebra	85
3.1	Basic Results from Linear Algebra	85
3.2	Normed Linear Spaces	88
3.3	Direct Methods for Solving Linear Systems	105
3.3.1	Gaussian Elimination	105
3.3.2	Symmetric, Positive Definite Matrices	117
3.3.3	Tridiagonal Matrices	119
3.3.4	Block Tridiagonal Matrices	120
3.3.5	Roundoff Error and Conditioning in Gaussian Elimination	121
3.3.6	Iterative Refinement	127
3.3.7	Interval Bounds	130
3.3.8	Orthogonal Decomposition (<i>QR</i> Decomposition)	132
3.4	Iterative Methods for Solving Linear Systems	141
3.4.1	The Jacobi Method	143
3.4.2	The Gauss–Seidel Method	143
3.4.3	Successive Overrelaxation	145
3.4.4	Convergence of the SOR, Jacobi, and Gauss–Seidel Methods	146
3.4.5	The Interval Gauss–Seidel Method	153
3.4.6	Graph–Theoretic Properties of Matrices	157
3.4.7	Convergence Rate of the SOR Method	159
3.4.8	Convergence of General Matrix Fixed Point Iterations	160
3.4.9	The Block SOR Method	161
3.4.10	The Conjugate Gradient Method	163
3.4.11	Iterative Methods for Matrix Inversion	167
3.4.12	Krylov Subspace Methods	169
3.5	The Singular Value Decomposition	174
3.6	Exercises	180
4	Approximation Theory	191
4.1	Introduction	191
4.2	Norms, Projections, Inner Product Spaces, and Orthogonalization in Function Spaces	191
4.2.1	Best Approximations	192
4.2.2	Best Approximations in Inner Product Spaces	198
4.2.3	The Gram–Schmidt Process (Formal Treatment)	201
4.3	Polynomial Approximation	205
4.3.1	The Weierstrass Approximation Theorem	205
4.3.2	Taylor Polynomial Approximations	209
4.3.3	Lagrange Interpolation	209
4.3.4	The Newton Form for the Interpolating Polynomial	213
4.3.5	Hermite Interpolation	220
4.3.6	Least Squares Polynomial Approximation	222
4.3.7	Error in Least Squares Polynomial Approximation	228

4.3.8	Minimax (Best Uniform) Approximation	229
4.3.9	Interval (Rigorous) Bounds on the Errors	236
4.4	Piecewise Polynomial Approximation	238
4.4.1	Piecewise Linear Interpolation	238
4.4.2	Cubic Spline Interpolation	243
4.4.3	Cubic Spline Interpolants	246
4.5	Trigonometric Approximation	249
4.5.1	Least Squares Trigonometric Approximation (Fourier Series)	249
4.5.2	Trigonometric Interpolation on a Finite Point Set (the FFT)	254
4.5.3	The FFT Procedure	257
4.6	Rational Approximation	259
4.6.1	Introduction	259
4.6.2	Best Rational Approximations in the Uniform Norm	259
4.6.3	Padé Approximation	259
4.6.4	Rational Interpolation	266
4.7	Wavelet Bases	267
4.7.1	Assumed Properties of the Scaling Function	268
4.7.2	The Wavelet Basis and the Scaling Function	272
4.7.3	Construction of Scaling Functions	276
4.8	Least Squares Approximation on a Finite Point Set	279
4.9	Exercises	284
5	Eigenvalue-Eigenvector Computation	291
5.1	Basic Results from Linear Algebra	291
5.2	The Power Method	297
5.3	The Inverse Power Method	303
5.4	Deflation	305
5.5	The QR Method	307
5.5.1	Reduction to Hessenberg or Tridiagonal Form	307
5.5.2	The QR Method with Origin Shifts	308
5.5.3	The Double QR Algorithm	312
5.6	Jacobi Diagonalization (Jacobi Method)	315
5.7	Simultaneous Iteration (Subspace Iteration)	318
5.8	Exercises	319
6	Numerical Differentiation and Integration	323
6.1	Numerical Differentiation	323
6.1.1	Derivation with Taylor's Theorem	323
6.1.2	Derivation via Lagrange Polynomial Representation	324
6.1.3	Error Analysis	325
6.2	Automatic (Computational) Differentiation	327
6.2.1	The Forward Mode	328
6.2.2	The Reverse Mode	330

6.2.3	Implementation of Automatic Differentiation	333
6.3	Numerical Integration	333
6.3.1	Introduction	333
6.3.2	Newton-Cotes Formulas	336
6.3.3	Gaussian Quadrature	343
6.3.4	Romberg Integration	353
6.3.5	Multiple Integrals, Singular Integrals, and Infinite In- tervals	364
6.3.6	Monte Carlo and Quasi-Monte Carlo Methods	369
6.3.7	Adaptive Quadrature	373
6.3.8	Interval Bounds	375
6.4	Exercises	376
7	Initial Value Problems for Ordinary Differential Equations	381
7.1	Introduction	381
7.2	Euler's method	385
7.3	Single-Step Methods: Taylor Series and Runge-Kutta	389
7.3.1	Order and Consistency of Euler's Method	391
7.3.2	Order and Consistency of the Midpoint Method	391
7.3.3	An Error Bound for Single-Step Methods	392
7.3.4	Higher-Order Methods	395
7.4	Error Control and the Runge-Kutta-Fehlberg Method	401
7.5	Multistep Methods	405
7.6	Predictor-Corrector Methods	416
7.7	Stiff Systems	418
7.7.1	Absolute Stability of Methods for Systems	421
7.7.2	Methods for Stiff Systems	422
7.8	Extrapolation Methods	426
7.9	Application to Parameter Estimation in Differential Equations	430
7.10	Exercises	431
8	Numerical Solution of Systems of Nonlinear Equations	439
8.1	Introduction and Fréchet Derivatives	439
8.2	Successive Approximation (Fixed Point Iteration) and the Con- traction Mapping Theorem	442
8.3	Newton's Method and Variations	447
8.3.1	Some Local Convergence Results for Newton's Method	447
8.3.2	Convergence Rate of Newton's Method	450
8.3.3	Example of Newton's Method	452
8.3.4	Local, Semilocal, and Global Convergence	454
8.3.5	The Newton-Kantorovich Theorem	454
8.3.6	A Global Convergence Result for Newton's Method	455
8.3.7	Practical Considerations	458
8.4	Multivariate Interval Newton Methods	463
8.4.1	The Nonlinear Interval Gauss-Seidel Method	465

8.4.2	The Multivariate Krawczyk Method	470
8.4.3	Using Interval Gaussian Elimination	472
8.4.4	Relationship to the Kantorovich Theorem	472
8.5	Quasi-Newton Methods (Broyden's Method)	473
8.5.1	Practicalities	479
8.6	Methods for Finding All Solutions	480
8.6.1	Homotopy Methods	480
8.6.2	Branch and Bound Methods	481
8.7	Exercises	482
9	Optimization	487
9.1	Local Optimization	489
9.1.1	Introduction to Unconstrained Local Optimization . .	489
9.1.2	Golden Section Search	490
9.1.3	Relationship to Nonlinear Systems	491
9.1.4	Steepest Descent	493
9.1.5	Quasi-Newton Methods for Minimization	495
9.1.6	The Nelder–Mead Simplex (Direct Search) Method . .	497
9.1.7	Software for Unconstrained Local Optimization . . .	501
9.2	Constrained Local Optimization	501
9.3	Constrained Optimization and Nonlinear Systems	501
9.4	Linear Programming	503
9.4.1	Converting to Standard Form	505
9.4.2	The Fundamental Theorem of Linear Programming . .	506
9.4.3	The Simplex Method	507
9.4.4	Proof of the Fundamental Theorem of Linear Program- ming	511
9.4.5	Degenerate Cases	512
9.4.6	Failure of Linear Programming Solvers	513
9.4.7	Software for Linear Programming	514
9.4.8	Quadratic Programming and Convex Programming . .	514
9.5	Dynamic Programming	515
9.6	Global (Nonconvex) Optimization	518
9.6.1	Genetic Algorithms	520
9.6.2	Simulated Annealing	523
9.6.3	Branch and Bound Algorithms	523
9.7	Exercises	531
10	Boundary-Value Problems and Integral Equations	535
10.1	Boundary-Value Problems	535
10.1.1	Shooting Method for Numerical Solution of (10.1) . .	536
10.1.2	Finite-Difference Methods	540
10.1.3	Galerkin Methods	544
10.2	Approximation of Integral Equations	554
10.2.1	Volterra Integral Equations of Second Kind	554

10.2.2 Fredholm Integral Equations of Second Kind	559
10.3 Exercises	566
A Solutions to Selected Exercises	571
References	591
Index	599

List of Figures

1.1	Illustration of the Intermediate Value Theorem.	1
1.2	An example floating point system: $\beta = 10$, $t = 1$, and $m = 1$	12
2.1	Example for a special case of the Intermediate Value Theorem (Theorem 2.1).	36
2.2	Graph of $e^x + x$ for Example 2.1.	37
2.3	Example for Remark 2.1 (when f does not change sign in $[a, b]$).	38
2.4	Example of Remark 2.6 (monotonic convergence of fixed point iteration).	46
2.5	Illustration of two iterations of Newton's method.	49
2.6	Example for Remark 2.9 (divergence of Newton's method). On the left, the sequence diverges; on the right, the sequence oscillates.	50
2.7	Geometric interpretation of the secant method.	59
2.8	Geometric interpretation of Müller's method.	64
2.9	Disc in which roots must lie.	71
2.10	Annulus in which roots must lie.	71
3.1	Directed graphs for irreducible and reducible matrices.	157
3.2	Cyclic diagrams.	158
4.1	p is the best approximation to $f \in V$	193
4.2	The projection is the best approximation. See Prop. 4.1.	201
4.3	Graph of a nonsmooth but continuous $f(x)$; see Remark 4.10.	206
4.4	The Chebyshev equi-oscillation property (Theorem 4.7).	218
4.5	A better approximation to f than p_n^*	233
4.6	$n + 1$ alternations of the best approximation error (Theorem 4.12).	235
4.7	An example of a piecewise linear function.	239
4.8	Graphs of the "hat" functions $\varphi_i(x)$	240
4.9	B-spline basis functions.	244
4.10	The graph of a trigonometric polynomial $p(x)$	254
4.11	The graph of a piecewise constant function f , as in Example 4.21.	269
4.12	$w(x) = \varphi(2x) - \varphi(2x - 1)$ in Example 4.25.	275
4.13	Illustration of a Daubechies scaling function $\varphi(x)$	278
4.14	Graph of a possible linear least squares fit $u(x) = \lambda_1 + \lambda_2 x$	279

5.1	Illustration of Gerschgorin discs for Example 5.1.	294
6.1	Illustration of the total error (roundoff plus truncation) bound in forward difference quotient approximation to f'	326
6.2	Illustration of composite numerical integration.	334
7.1	Illustration of rounding plus discretization error (Remark 7.5).	389
7.2	Sudden increase in a solution.	401
7.3	Local error in integrating an ODE.	402
7.4	Exact solution for the stiff ODE system given in Example 7.10.	420
7.5	The open disc representing the stability region for Euler's method.	422
7.6	Illustration of the region of absolute stability for $A(0)$ -stability (Definition 7.16).	425
9.1	A local minimizer x^* only minimizes over some $\bar{S}$	489
9.2	Illustration of 2- and 3-dimensional simplexes.	499
9.3	Graph of the constraint set and objective function z , Example 9.4.	504
9.4	Graph of the constraint set and the objective function z for Example 9.8, an unbounded LP.	513
9.5	Diagram of the fish population for Example 9.10 of a multistage decision process.	516
9.6	Profit diagram for Example 9.10 of a multistage decision process.	517
9.7	The search tree for Example 9.13	525
10.1	Illustration of the shooting method.	539
10.2	A finite difference mesh	540

List of Tables

1.1	Parameters for IEEE arithmetic	19
1.2	Machine constants for IEEE arithmetic	20
2.1	Convergence of the interval Newton method with $f(x) = x^2 - 2$.	58
2.2	A summary of the methods considered in Chapter 2.	84
3.1	Condition numbers of some Hilbert matrices	125
3.2	Iterates of the Jacobi and Gauss–Seidel methods, for Example 3.11	145
4.1	Legendre polynomial approximation of degree 3 to $\sin(\pi x/2)$	225
4.2	Error factors K and M in polynomial approximations $f(x) = p(x) + K(x)M(f; x)$	236
4.3	Comparison of the Padé approximant of Example 4.14 to a degree-2 Taylor polynomial	263
4.4	Padé approximants to e^x	265
6.1	Weights and sample points: Gauss–Legendre quadrature . . .	349
6.2	Illustration of Richardson extrapolation	354
6.3	Illustration of even-order Richardson extrapolation	354
6.4	Romberg integration and composite Newton–Cotes methods .	361
6.5	Subtracting out the singularity, Example 2	367
7.1	Padé approximations to e^z	425

Chapter 1

Mathematical Review and Computer Arithmetic

1.1 Mathematical Review

In this section, several mathematical results and definitions are reviewed that will be useful throughout many of the following chapters.

1.1.1 Intermediate Value Theorem, Mean Value Theorems, and Taylor's Theorem

Throughout, $C^n[a, b]$ will denote the set of real-valued functions f defined on the interval $[a, b]$ such that f and its derivatives, up to and including its n -th derivative $f^{(n)}$, are continuous on $[a, b]$.

THEOREM 1.1

(Intermediate value theorem) If $f \in C[a, b]$ and k is any number between $m = \min_{a \leq x \leq b} f(x)$ and $M = \max_{a \leq x \leq b} f(x)$, then there exists a number c in $[a, b]$ for which $f(c) = k$ (Figure 1.1).

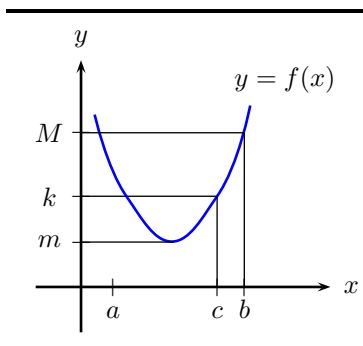


FIGURE 1.1: Illustration of the Intermediate Value Theorem.

THEOREM 1.2

(Mean value theorem for integrals) Let f be continuous and w be Riemann integrable¹ on $[a, b]$ and suppose that $w(x) \geq 0$ for $x \in [a, b]$. Then there exists a point ξ in $[a, b]$ such that

$$\int_a^b w(x)f(x)dx = f(\xi) \int_a^b w(x)dx.$$

PROOF Let $A = \min_{a \leq x \leq b} f(x)$ and $B = \max_{a \leq x \leq b} f(x)$. Then $Aw(x) \leq f(x)w(x) \leq Bw(x)$. Hence,

$$A \int_a^b w(x)dx \leq \int_a^b w(x)f(x)dx \leq B \int_a^b w(x)dx.$$

Thus,

$$A \leq \frac{\int_a^b w(x)f(x)dx}{\int_a^b w(x)dx} \leq B.$$

By the Intermediate Value Theorem, there is a ξ in $[a, b]$ such that

$$f(\xi) = \frac{\int_a^b w(x)f(x)dx}{\int_a^b w(x)dx}.$$

□

THEOREM 1.3

(Taylor's theorem) Suppose that $f \in C^{n+1}[a, b]$. Let $x_0 \in [a, b]$. Then for any $x \in [a, b]$,

$f(x) = P_n(x) + R_n(x)$, where

$$P_n(x) = f(x_0) + f'(x_0)(x - x_0) + \cdots + \frac{f^{(n)}(x_0)(x - x_0)^n}{n!}$$

$$= \sum_{k=0}^n \frac{1}{k!} f^{(k)}(x_0)(x - x_0)^k, \text{ and}$$

$$R_n(x) = \frac{1}{n!} \int_{x_0}^x f^{(n+1)}(t)(x - t)^n dt \text{ (integral form of remainder).}$$

Furthermore, there is a $\xi = \xi(x)$ between x_0 and x with

$$R_n(x) = \frac{f^{(n+1)}(\xi(x))(x - x_0)^{n+1}}{(n + 1)!} \text{ (Lagrange form of remainder).}$$

¹In most, but not all contexts in numerical analysis, w will be continuous.

PROOF Recall the integration by parts formula $\int u dv = uv - \int v du$. Thus,

$$\begin{aligned}
 f(x) - f(x_0) &= \int_{x_0}^x f'(t) dt \quad (\text{let } u = f'(t), v = t - x, dv = dt) \\
 &= f'(x_0)(x - x_0) + \int_{x_0}^x (x - t) f''(t) dt \\
 &\quad (\text{let } u = f''(t), dv = (x - t) dt) \\
 &= f'(x_0)(x - x_0) - \frac{(x - t)^2}{2} f''(t) \Big|_{x_0}^x + \int_{x_0}^x \frac{(x - t)^2}{2} f'''(t) dt \\
 &= f'(x_0)(x - x_0) + \frac{(x - x_0)^2}{2} f''(x_0) + \int_{x_0}^x \frac{(x - t)^2}{2} f'''(t) dt
 \end{aligned}$$

Continuing this procedure,

$$\begin{aligned}
 f(x) &= f(x_0) + f'(x_0)(x - x_0) + \frac{(x - x_0)^2}{2} f''(x_0) \\
 &\quad + \cdots + \frac{(x - x_0)^n}{n!} f^{(n)}(x_0) + \int_{x_0}^x \frac{(x - t)^n}{n!} f^{(n+1)}(t) dt \\
 &= P_n(x) + R_n(x)
 \end{aligned}$$

Now consider $R_n(x) = \int_{x_0}^x \frac{(x - t)^n}{n!} f^{(n+1)}(t) dt$ and assume that $x_0 < x$ (same argument if $x_0 > x$). Then, by Theorem 1.2,

$$R_n(x) = f^{(n+1)}(\xi(x)) \int_{x_0}^x \frac{(x - t)^n}{n!} dt = f^{(n+1)}(\xi(x)) \frac{(x - x_0)^{n+1}}{(n + 1)!},$$

where ξ is between x_0 and x and thus, $\xi = \xi(x)$. □

An important special case of Taylor's theorem is obtained with $n = 0$ (that is, directly from the Fundamental Theorem of Calculus).

THEOREM 1.4

(Mean value theorem) Suppose $f \in C^1[a, b]$, $x \in [a, b]$, and $y \in [a, b]$ (and, without loss of generality, $x \leq y$). Then there is a $\xi \in [x, y] \subseteq [a, b]$ such that

$$f(y) - f(x) = f'(\xi)(y - x).$$

Example 1.1

Show that $e^x \geq 1 + x + \frac{x^2}{2}$ for all $x \geq 0$. □

PROOF By Taylor's Theorem,

$$e^x = f(0) + f'(0)(x-0) + f''(0)\frac{(x-0)^2}{2} + \int_0^x \frac{(x-t)^2 f'''(t) dt}{2},$$

where $f(x) = e^x$. Thus,

$$e^x = 1 + x + \frac{x^2}{2} + \int_0^x \frac{1}{2}(x-t)^2 e^t dt,$$

and it follows that $e^x \geq 1 + x + \frac{x^2}{2}$ for $x \geq 0$, since

$$\int_0^x \frac{1}{2}(x-t)^2 e^t dt \geq 0 \text{ for } x \geq 0.$$

□

Example 1.2

Show that $\left| \frac{f(x+h)-f(x)}{h} - f'(x) \right| \leq ch$ for $x, x+h \in [a, b]$, assuming that $f \in C^2[a, b]$. □

PROOF

$$\begin{aligned} & \left| \frac{f(x+h)-f(x)}{h} - f'(x) \right| \\ &= \left| \frac{f(x) + f'(x)h + \int_x^{x+h} (x+h-t)f''(t)dt - f(x)}{h} - f'(x) \right| \\ &= \frac{1}{h} \left| \int_x^{x+h} (x+h-t)f''(t)dt \right| \leq \max_{a \leq t \leq b} |f''(t)| \frac{h}{2} = ch. \end{aligned}$$

□

THEOREM 1.5

(Taylor's Theorem in Two Variables) Suppose that $f(x, y)$ and all its partial derivatives of order less than or equal to $n+1$ are continuous in the rectangle $D = \{(x, y) | a \leq x \leq b, c \leq y \leq d\}$. Let $(x_0, y_0) \in D$. For every $(x, y) \in D$,

there exists a ξ between x and x_0 and an η between y and y_0 such that

$$\begin{aligned} f(x, y) &= P_n(x, y) + R_n(x, y) \text{ where} \\ P_n(x, y) &= f(x_0, y_0) + \left[(x - x_0) \frac{\partial f}{\partial x}(x_0, y_0) + (y - y_0) \frac{\partial f}{\partial y}(x_0, y_0) \right] + \cdots + \\ &\quad \frac{1}{n!} \left[\sum_{j=0}^n \binom{n}{j} (x - x_0)^{n-j} (y - y_0)^j \frac{\partial^n f(x_0, y_0)}{\partial x^{n-j} \partial y^j} \right] \text{ and} \\ R_n(x, y) &= \frac{1}{(n+1)!} \left[\sum_{j=0}^{n+1} \binom{n+1}{j} (x - x_0)^{n+1-j} (y - y_0)^j \frac{\partial^{n+1} f(\xi, \eta)}{\partial x^{n+1-j} \partial y^j} \right]. \end{aligned}$$

1.1.2 Big “O” and Little “o” Notation

We study “rates of growth” and “rates of decrease” of errors. For example, if we approximate e^h by a first degree Taylor polynomial about $x = 0$, we get

$$e^h - (1 + h) = \frac{1}{2} h^2 e^\xi,$$

where ξ is some unknown quantity between 0 and h . Although we don’t know exactly what e^ξ is, we know that it is nearly constant (in this case, approximately 1) for h near 0, so the error $e^h - (1 + h)$ is roughly proportional to h^2 for h small. This approximate proportionality is often more important to know than the slowly-varying constant e^ξ . The big “O” and little “o” notation are used to describe and keep track of this approximate proportionality.

DEFINITION 1.1 (*Big O and little o notation*) Let $\{x_k\}$ and $\{\alpha_k\}$ be two sequences. We write $x_k = \mathcal{O}(\alpha_k)$ as $k \rightarrow \infty$ if there are constants c and r such that $|x_k| \leq c|\alpha_k|$ when $k \geq r$. We write $x_k = o(\alpha_k)$ if, given any $\epsilon > 0$, there is an r such that $|x_k| < \epsilon|\alpha_k|$ for $k \geq r$. Similarly, if $f(h)$ and $g(h)$ are functions of a continuous variable h , we say that $f(h) = \mathcal{O}(g(h))$ as $h \rightarrow 0$ provided there are constants c and δ such that $|f(h)| \leq c|g(h)|$ for $|h| < \delta$.

Note that

$$\text{if } x_k = \mathcal{O}(\alpha_k), \text{ then } \lim_{k \rightarrow \infty} \left| \frac{x_k}{\alpha_k} \right| \leq c.$$

Similarly,

$$\text{if } x_k = o(\alpha_k), \text{ then } \lim_{k \rightarrow \infty} \left| \frac{x_k}{\alpha_k} \right| = 0.$$

The “O” denotes “order.” For example, if $f(h) = \mathcal{O}(h^2)$, we say that “ f exhibits order 2 convergence to 0 as h tends to 0.”

REMARK 1.1 The “big $\mathcal{O}$ ” notation is often used in the context of error analysis for vector-valued functions. Let $e(h)$ be a vector that represents an error in some other vector; we then might say

$$e(h) = \mathcal{O}(h^k) \quad \text{as } h \rightarrow 0,$$

provided there exists a constant c independent of h , such that $\|e(h)\| \leq ch^k$ for all $h \in [0, h_0]$. (Roughly, for sufficiently small h , $\|e(h)\|$ does not go to zero slower than h^k .) $\square$

Example 1.3

$x_k = e^{1/k} - 1/k - 1$, $\alpha_k = \frac{1}{k^2}$. Then $x_k = \mathcal{O}(\alpha_k)$. $\square$

PROOF By Taylor’s Theorem,

$$e^{1/k} = e^0 + e'(0)(1/k - 0) + \int_0^{1/k} e^t(1/k - t) dt.$$

Thus, $e^{1/k} - 1 - 1/k \leq \frac{1}{k^2} \left(\frac{e^1}{2} \right)$ for $k \geq 1$. $\square$

The following definition extends this notation to functions.

DEFINITION 1.2 Suppose that $\lim_{h \rightarrow 0} f(h) = L$. We write $f(h) - L = \mathcal{O}(g(h))$ if there is a constant K such that $\frac{|f(h) - L|}{|g(h)|} < K$ for sufficiently small $h > 0$.

Example 1.4

Show that $\cos h - 1 + \frac{h^2}{2} = \mathcal{O}(h^4)$. $\square$

PROOF By Taylor’s Theorem,

$$\cos h = 1 - \frac{h^2}{2} + \frac{1}{3!} \int_0^h \cos(t)(h - t)^3 dt.$$

Thus,

$$\left| \cos h - 1 + \frac{h^2}{2} \right| \leq \frac{1}{6} \int_0^h (h - t)^3 dt = \frac{h^4}{24}.$$

Hence,

$$\frac{|\cos h - 1 + \frac{h^2}{2}|}{|h^4|} \leq \frac{1}{24}.$$

$\square$

1.1.3 Convergence Rates

DEFINITION 1.3 Let $\{x_k\}$ be a sequence with limit $\mathbf{x}^*$. If there are constants C and α and an integer N such that $|x_{k+1} - \mathbf{x}^*| \leq C|x_k - \mathbf{x}^*|^\alpha$ for $k \geq N$ we say that the rate of convergence is of order at least α . If $\alpha = 1$ (with $C < 1$), the rate is said to be linear. If $\alpha = 2$, the rate is said to be quadratic.

Example 1.5

Show that the sequence $\{x_k\}$ defined by $x_{k+1} = \frac{x_k^2 + 4}{2x_k}$, $k = 0, 1, 2, \dots$, with $x_0 = 3$, converges quadratically to $\mathbf{x}^* = 2$. $\square$

PROOF

(Part 1: showing convergence) It is first shown that $\lim_{k \rightarrow \infty} x_k = 2$. If $x_k > 2$, then $x_k^2 - 4 > 0$. Thus, $2x_k^2 > x_k^2 + 4$. Hence, it follows that

$$x_k > \frac{x_k^2 + 4}{2x_k} = x_{k+1}. \quad (1.1)$$

Combining this with $-(x_k - 2)^2 < 0$ gives $4x_k - x_k^2 - 4 < 0$. Hence, $4x_k < x_k^2 + 4$ and thus,

$$2 < \frac{x_k^2 + 4}{2x_k} = x_{k+1}. \quad (1.2)$$

By (1.1) and (1.2), $x_k > x_{k+1} > 2$. Hence,

$$x_0 > x_1 > x_2 > x_3 > \dots > x_k > \dots > 2.$$

Therefore, $\{x_k\}$ is a monotonically decreasing sequence bounded below by 2 and thus has a limit $\mathbf{x}^*$. But $\mathbf{x}^* = \frac{(\mathbf{x}^*)^2 + 4}{2\mathbf{x}^*}$ gives $\mathbf{x}^* = 2$.

(Part 2: showing that the convergence is quadratic) We see that

$$\begin{aligned} |x_{k+1} - 2| &= \left| \frac{x_k^2 + 4}{2x_k} - 2 \right| = \left| \frac{x_k^2 + 4 - 4x_k}{2x_k} \right| \\ &= \left| \frac{(x_k - 2)^2}{2x_k} \right| \leq \frac{1}{4} |x_k - 2|^2, \end{aligned}$$

since $x_k \geq 2$. Therefore, $\{x_k\}$ converges quadratically to 2. $\square$

Quadratic convergence is very fast. For the above example, if $|x_k - 2| = 0.01$ then $|x_{k+1} - 2| \leq \frac{1}{4}(0.01)^2$, $|x_{k+2} - 2| \leq \frac{1}{64}(0.01)^4$, We give some computational results for this example with $x_0 = 3$:

k	x_k
0	3
1	2.16667
2	2.00641
3	2.00001
4	2.00000

Sometimes, it is not practical to devise a computation to exhibit quadratic convergence. In such cases, however, a convergence rate that is almost as fast can sometimes be achieved:

DEFINITION 1.4 *A sequence $\{x_k\}$ with limit x^* is said to exhibit superlinear convergence, provided*

$$\frac{|x_{k+1} - x^*|}{|x_k - x^*|} \rightarrow 0 \text{ as } k \rightarrow \infty.$$

Superlinear convergence is faster than linear, but can, in principle, be slower than convergence of order α , for any $\alpha > 1$.

Example 1.6

The sequence $\{x_k\}$ defined by

$$x_k = \frac{1}{2^{k^2}}$$

is superlinearly convergent to 0, since

$$\frac{|x_{k+1} - 0|}{|x_k - 0|} = \frac{2^{k^2}}{2^{(k+1)^2}} = \left(\frac{1}{2}\right) \left(\frac{1}{2^{2k}}\right) \rightarrow 0 \text{ as } k \rightarrow \infty$$

However, $\{x_k\}$ is not quadratically convergent, since

$$\frac{|x_{k+1} - 0|}{|x_k - 0|^2} = \frac{1/2^{(k+1)^2}}{1/(2^{k^2})^2} = 2^{k^2 - 2k - 1} \not\rightarrow 0 \text{ as } k \rightarrow \infty.$$

In fact, it can be shown that $\{x_k\}$ is not convergent with convergence order α for any $\alpha > 1$. $\square$

1.2 Computer Arithmetic

In numerical solution of mathematical problems, two common types of error are:

1. Method (algorithm or truncation) error. This is the error due to approximations made in the numerical method.
2. Rounding error. This is the error made due to the finite number of digits available on a computer.

Example 1.7

By the mean value theorem for integrals (Theorem 1.2, as in Example 1.2 on page 4), if $f \in C^2[a, b]$, then

$$f'(x) = \frac{f(x+h) - f(x)}{h} + \frac{1}{h} \int_x^{x+h} f''(t)(x+h-t)dt$$

$$\text{and } \left| \frac{1}{h} \int_x^{x+h} f''(t)(x+h-t)dt \right| \leq ch.$$

Thus, $f'(x) \approx (f(x+h) - f(x))/h$, and the error is $\mathcal{O}(h)$. We will call this the *method error* or *truncation error*, as opposed to *roundoff errors* due to using machine approximations. $\square$

Now consider $f(x) = \ln x$ and approximate $f'(3) \approx \frac{\ln(3+h) - \ln 3}{h}$ for h small using a calculator having 11 digits. The following results were obtained.

h	$\frac{\ln(3+h) - \ln(3)}{h}$	Error = $\frac{1}{3} - \frac{\ln(3+h) - \ln(3)}{h} = \mathcal{O}(h)$
10^{-1}	0.3278982	5.44×10^{-3}
10^{-2}	0.332779	5.54×10^{-4}
10^{-3}	0.3332778	5.55×10^{-5}
10^{-4}	0.333328	5.33×10^{-6}
10^{-5}	0.333330	3.33×10^{-6}
10^{-6}	0.333300	3.33×10^{-5}
10^{-7}	0.333	3.33×10^{-4}
10^{-8}	0.33	3.33×10^{-3}
10^{-9}	0.3	3.33×10^{-2}
10^{-10}	0.0	3.33×10^{-1}

One sees that, in the first four steps, the error decreases by a factor of 10 as h is decreased by a factor of 10 (That is, the *method error* dominates). However, starting with $h = 0.00001$, the error increases. (The error due to a finite number of digits, i.e., *roundoff error* dominates).

REMARK 1.2 Problems with round-off error have contributed to several major disasters in the real-world. One such example is the Patriot Missile failure, in Dharan, Saudi Arabia, on February 25, 1991. This incident resulted

in 28 deaths, and was ultimately attributable to poor handling of roundoff error. $\square$

REMARK 1.3 There are two possible ways to reduce rounding error:

1. The method error can be reduced by using a more accurate method. This allows larger h to be used, thus avoiding roundoff error. Consider

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + \{\text{error}\}, \text{ where } \{\text{error}\} \text{ is } \mathcal{O}(h^2).$$

h	$\frac{\ln(3+h) - \ln(3-h)}{2h}$	error
0.1	0.3334568	1.24×10^{-4}
0.01	0.3333345	1.23×10^{-6}
0.001	0.3333333	1.91×10^{-8}

The error decreases by a factor of 100 as h is decreased by a factor of 10.

2. Rounding error can be reduced by using more digits of accuracy, such as using double precision (or multiple precision) arithmetic.

$\square$

To fully understand and avoid roundoff error, we should study some details of how computers and calculators represent and work with approximate numbers.

1.2.1 Floating Point Arithmetic and Rounding Error

Let $\beta = \{\text{a positive integer}\}$, the base of the computer system. (Usually, $\beta = 2$ (binary) or $\beta = 16$ (hexadecimal)). Suppose a number x has the exact base representation

$$x = (\pm 0.\alpha_1\alpha_2\alpha_3 \cdots \alpha_t\alpha_{t+1} \cdots)\beta^m = \pm q\beta^m,$$

where q is the *mantissa*, β is the base, m is the exponent, $1 \leq \alpha_1 \leq \beta - 1$ and $0 \leq \alpha_i \leq \beta - 1$ for $i > 1$.

On a computer, we are restricted to a finite set of *floating-point numbers* $F = F(\beta, t, L, U)$ of the form $\mathbf{x}^* = (\pm 0.a_1a_2 \cdots a_t)\beta^m$, where $1 \leq a_1 \leq \beta - 1$, $0 \leq a_i \leq \beta - 1$ for $2 \leq i \leq t$, $L \leq m \leq U$, and t is the number of digits. (In most floating point systems, L is about -64 to -1000 and U is about 64 to 1000 .)

Example 1.8(binary) $\beta = 2$

$$\begin{aligned} \mathbf{x}^* &= (0.1011)2^3 = \left(1 \times \frac{1}{2} + 0 \times \frac{1}{4} + 1 \times \frac{1}{8} + 1 \times \frac{1}{16}\right) \times 8 \\ &= \frac{11}{2} = 5.5 \text{ (decimal)}. \end{aligned}$$

□

REMARK 1.4 Most numbers cannot be exactly represented on a computer. Consider $x = 10.1 = 1010.0001\ 1001\ \overline{1001}$ ($\beta = 2$). If $L = -127$, $U = 127$, $t = 24$, and $\beta = 2$, then $x \approx \mathbf{x}^* = (0.10100001\ 1001\ 1001\ 1001\ 1001)2^4$. □

Question: Given a real number x , how do we define a floating point number $\text{fl}(x)$ in F , such that $\text{fl}(x)$ is close to x ?

Here are two possible procedures² for choosing $\text{fl}(x)$:

1. *Chop:* $\text{fl}(x)$ is that element of F such that $\text{fl}(x) = \text{sgn}(x)(0.a_1a_2 \cdots a_t)\beta^m$, where $a_i = \alpha_i$, $1 \leq i \leq t$, where $|x|$ has infinite base β expansion

$$|x| = (0.a_1a_2 \cdots a_ta_{t+1} \cdots) \times \beta^m, \quad (1.3)$$

and where

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x \geq 0, \\ -1 & \text{if } x < 0. \end{cases} \quad (1.4)$$

2. *Round:* $\text{fl}(x)$ is that element of F closest to x , and if x is exactly between two elements of F , then $\text{fl}(x)$ is generally taken to be the element of largest magnitude. Thus, for round, if $|x|$ has base β expansion as in (1.3), let

$$|x| + \frac{1}{2}\beta^{-t}\beta^m = (0.a_1^*a_2^* \cdots a_t^*a_{t+1}^* \cdots)\beta^m. \quad (1.5)$$

Then, $\text{fl}(x) = \text{sgn}(x)(0.a_1^*a_2^* \cdots a_t^*)\beta^m$.

Example 1.9

$\beta = 10$, $t = 5$, $x = 0.12345666 \cdots \times 10^7$. Then

$$\text{fl}(x) = 0.12345 \times 10^7 \text{ (chopped).}$$

$$\text{fl}(x) = 0.12346 \times 10^7 \text{ (rounded).}$$

²On most modern machines, four rounding modes are actually chosen. See Section 1.2.2, starting on page 17.

(Note: $x + \frac{1}{2}\beta^{-t}\beta^m = 0.12345666 \dots \times 10^7 + \frac{1}{2}10^{-5}10^7 = 0.12346166 \dots \times 10^7$.)

□

See Figure 1.2 for an example with $\beta = 10$ and $t = 1$. In that figure, the exhibited floating point numbers are $(0.1) \times 10^1$, $(0.2) \times 10^1$, $\dots$, $(0.9) \times 10^1$, 0.1×10^2 .

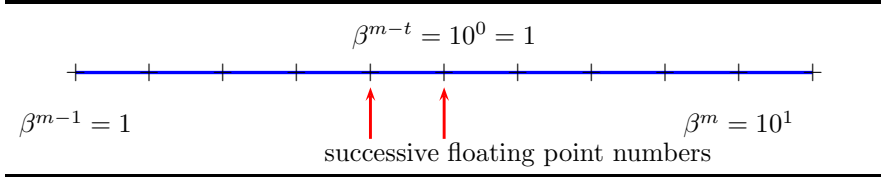


FIGURE 1.2: An example floating point system: $\beta = 10$, $t = 1$, and $m = 1$.

We now have the following error bound.

THEOREM 1.6

$$|x - f(x)| \leq \frac{1}{2}|x|\beta^{1-t}p,$$

where $p = 1$ for rounding and $p = 2$ for chopping.

PROOF Since $x = (\pm 0.\alpha_1\alpha_2 \dots \alpha_t \dots)\beta^m$, we have $\beta^{m-1} \leq |x| \leq \beta^m$. In the interval $[\beta^{m-1}, \beta^m]$, the floating point numbers are evenly spaced with separation β^{m-t} . Thus, for chopping,

$$|x - f(x)| \leq \beta^{m-t} = \frac{p}{2}\beta^{m-t},$$

and for rounding,

$$|x - f(x)| \leq \frac{1}{2}\beta^{m-t} = \frac{p}{2}\beta^{m-t}.$$

Hence,

$$|x - f(x)| \leq \frac{p}{2}\beta^{m-t} \leq \frac{p}{2}\beta^{1-t}\beta^{m-1} \leq \frac{1}{2}|x|\beta^{1-t}p.$$

□

REMARK 1.5 $\delta = \frac{p}{2}\beta^{1-t}$ is called the unit roundoff error.

□

REMARK 1.6 Let $\epsilon = \frac{fl(x) - x}{x}$. Then $fl(x) = (1 + \epsilon)x$, where $|\epsilon| \leq \delta$.

□

Now consider errors produced in the arithmetic operations $x + y$, $x - y$, xy , and x/y .

THEOREM 1.7

Let $\odot$ denote the operation $+$, $-$, $\times$, or $\div$, and let x and y be machine numbers. Then

$$fl(x \odot y) = (x \odot y)(1 + \epsilon), \text{ where } |\epsilon| \leq \delta = \frac{p}{2} \beta^{1-t}.$$

PROOF Theorem 1.6 gives

$$|x \odot y - fl(x \odot y)| \leq |x \odot y| \frac{p}{2} \beta^{1-t}.$$

Thus,

$$-|x \odot y| \frac{p}{2} \beta^{1-t} \leq -x \odot y + fl(x \odot y) \leq \frac{p}{2} \beta^{1-t} |x \odot y|.$$

Hence,

$$(x \odot y) \left(1 - \frac{|x \odot y|}{x \odot y} \frac{p}{2} \beta^{1-t} \right) \leq fl(x \odot y) \leq (x \odot y) \left(1 + \frac{|x \odot y|}{x \odot y} \frac{p}{2} \beta^{1-t} \right).$$

It follows that $fl(x \odot y) = (1 + \epsilon)(x \odot y)$ where $|\epsilon| \leq \frac{p}{2} \beta^{1-t} = \delta$. □

Example 1.10

$\beta = 10$, $t = 4$, $p = 1$. (Thus, $\delta = \frac{1}{2}10^{-3} = 0.0005$.) Let $x = 0.5795 \times 10^5$, $y = 0.6399 \times 10^5$. Then

$$\begin{aligned} fl(x + y) &= 0.1219 \times 10^6 = (x + y)(1 + \epsilon_1), \quad \epsilon_1 = 0.00033 < \delta, \quad \text{and} \\ fl(xy) &= 0.3708 \times 10^{10} = (xy)(1 + \epsilon_2), \quad \epsilon_2 = 0.000059 < \delta. \end{aligned}$$

(Note: $x + y = 0.12194 \times 10^6$, $xy = 0.37082205 \times 10^{10}$.) □

Let's apply our result to a more complicated problem. Let x_1 and x_2 be two exact numbers. Consider

$$\begin{aligned} fl(x_1 + x_2) &= fl[fl(x_1) + fl(x_2)] = fl[x_1(1 + \hat{\epsilon}_1) + x_2(1 + \hat{\epsilon}_2)] \\ &= [x_1(1 + \hat{\epsilon}_1) + x_2(1 + \hat{\epsilon}_2)](1 + \epsilon_1), \end{aligned}$$

where $|\hat{\epsilon}_1| \leq \delta$, $|\hat{\epsilon}_2| \leq \delta$, and $|\epsilon_1| \leq \delta$. Similarly,

$$\begin{aligned} fl(x_1 + x_2 + x_3) &= ((x_1(1 + \hat{\epsilon}_1) + x_2(1 + \hat{\epsilon}_2))(1 + \epsilon_1) + x_3(1 + \hat{\epsilon}_3))(1 + \epsilon_2) \\ &= x_1(1 + \hat{\epsilon}_1)(1 + \epsilon_1)(1 + \epsilon_2) + x_2(1 + \hat{\epsilon}_2)(1 + \epsilon_1)(1 + \epsilon_2) \\ &\quad + x_3(1 + \hat{\epsilon}_3)(1 + \epsilon_2) \end{aligned}$$

Continuing this procedure,

$$\begin{aligned}
 fl\left(\sum_{i=1}^n x_i\right) &= x_1(1 + \hat{\epsilon}_1) \prod_{i=1}^{n-1} (1 + \epsilon_i) \\
 &\quad + x_2(1 + \hat{\epsilon}_2) \prod_{i=1}^{n-1} (1 + \epsilon_i) + x_3(1 + \hat{\epsilon}_3) \prod_{i=2}^{n-1} (1 + \epsilon_i) \\
 &\quad + x_4(1 + \hat{\epsilon}_4) \prod_{i=3}^{n-1} (1 + \epsilon_i) + \cdots + x_n(1 + \hat{\epsilon}_n)(1 + \epsilon_{n-1}).
 \end{aligned}$$

Considering this expression, it is clear that, to reduce rounding error, a sequence should be summed from small numbers to large numbers on a computer.

Example 1.11

Suppose $\beta = 10$ and $t = 4$ (4 digit arithmetic), suppose $x_1 = 10000$ and $x_2 = x_3 = \cdots = x_{1001} = 1$. Then

$$\begin{aligned}
 fl(x_1 + x_2) &= 10000, \\
 fl(x_1 + x_2 + x_3) &= 10000, \\
 &\vdots \\
 fl\left(\sum_{i=1}^{1001} x_i\right) &= 10000,
 \end{aligned}$$

when we sum forward from x_1 . But going backwards,

$$\begin{aligned}
 fl(x_{1001} + x_{1000}) &= 2, \\
 fl(x_{1001} + x_{1000} + x_{999}) &= 3, \\
 &\vdots \\
 fl\left(\sum_{i=1001}^1 x_i\right) &= 11000,
 \end{aligned}$$

which is the correct sum. □

We now have some useful definitions.

DEFINITION 1.5 Let $\mathbf{x}^*$ be an approximation to x . Then $|x - \mathbf{x}^*|$ is called the absolute error, and $\left|\frac{x - \mathbf{x}^*}{x}\right|$ is called the relative error.

For example, $\left| \frac{x - fl(x)}{x} \right| \leq \delta = \frac{p}{2} \beta^{1-t}$ (unit roundoff error).

REMARK 1.7 Large relative errors can occur when two nearly equal numbers are subtracted on a computer. $\square$

Example 1.12

$x_1 = 15.314768$, $x_2 = 15.314899$, $\beta = 10$, $t = 6$ (6-digit decimal accuracy). Then $x_2 - x_1 \approx fl(x_2) - fl(x_1) = 15.3149 - 15.3148 = 0.0001$. Thus,

$$\begin{aligned} \left| \frac{x_2 - x_1 - (fl(x_2) - fl(x_1))}{x_2 - x_1} \right| &= \frac{0.000131 - 0.0001}{0.000131} \\ &= 0.237 \\ &= 23.7\% \text{ relative accuracy.} \end{aligned}$$

$\square$

REMARK 1.8 Sometimes, an algorithm can be modified to reduce rounding error, such as when the rounding error is caused by subtraction of nearly equal quantities. $\square$

Example 1.13

Consider finding the roots of $ax^2 + bx + c = 0$, where b^2 is large compared with $|4ac|$. The most common formula for the roots is

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

Consider $x^2 + 100x + 1 = 0$, $\beta = 10$, $t = 4$, $p = 2$, and 4-digit chopped arithmetic. Then

$$x_1 = \frac{-100 + \sqrt{9996}}{2}, \quad x_2 = \frac{-100 - \sqrt{9996}}{2},$$

but $\sqrt{9996} \approx 99.97$ (4 digit arithmetic chopped). Thus,

$$x_1 \approx \frac{-100 + 99.97}{2}, \quad x_2 \approx \frac{-100 - 99.97}{2}.$$

Hence, $x_1 \approx -0.015$, $x_2 \approx -99.98$, but $x_1 = -0.010001$ and $x_2 = -99.989999$, so the relative errors in x_1 and x_2 are 50% and 0.01%, respectively.

Let's change the algorithm. Assume $b \geq 0$ (can always make $b \geq 0$). Then

$$\begin{aligned} x_1 &= \frac{-b + \sqrt{b^2 - 4ac}}{2a} \left(\frac{-b - \sqrt{b^2 - 4ac}}{-b - \sqrt{b^2 - 4ac}} \right) \\ &= \frac{4ac}{2a(-b - \sqrt{b^2 - 4ac})} = \frac{-2c}{b + \sqrt{b^2 - 4ac}}, \\ \text{and} \\ x_2 &= \frac{-b - \sqrt{b^2 - 4ac}}{2a} \quad (\text{the same as before}). \end{aligned}$$

Then, for the above values,

$$x_1 = \frac{-2(1)}{100 + \sqrt{9996}} \approx \frac{-2}{100 + 99.97} = -0.0100.$$

Now, the relative error in x_1 is also 0.01%. □

Let us now consider error in function evaluation. Consider a single valued function $f(x)$ and let $x^* = fl(x)$ be the floating point approximation of x . Therefore the machine evaluates $f(x^*) = f(fl(x))$, which is an approximate value of $f(x)$ at $x = x^*$. Then the perturbation in $f(x)$ for small perturbations in x can be computed via Taylor's formula. This is illustrated in the next theorem.

THEOREM 1.8

The relative error in functional evaluation is,

$$\left| \frac{f(x) - f(x^*)}{f(x)} \right| \approx \left| \frac{x f'(x^*)}{f(x)} \right| \left| \frac{x - x^*}{x} \right|$$

PROOF The linear Taylor approximation of $f(x)$ about $f(x^*)$ for small values of $|x - x^*|$ is given by $f(x) \approx f(x^*) + f'(x^*)(x - x^*)$. Rearranging the terms immediately yields the result. □

We now define the *condition number* of a function $f(x)$ as

$$\kappa_f(x^*) := \left| \frac{x f'(x^*)}{f(x)} \right|$$

which describes how large the relative error in function evaluation is with respect to the relative error in the machine representation of x . In other words, $\kappa_f(x^*)$ is a measure of the degree of sensitivity of the function at $x = x^*$.

Example 1.14

Let $f(x) = \sqrt{x}$. The condition number of $f(x)$ about $x = x^*$ is

$$\kappa_f(x^*) = \left| \frac{x \frac{1}{2\sqrt{x}}}{\sqrt{x}} \right|_{x=x^*} = \frac{1}{2}.$$

This suggests that $f(x)$ is well-conditioned. □

Example 1.15

Let $f(x) = \sqrt{x-2}$. The condition number of $f(x)$ about $x = x^*$ is

$$\kappa_f(x^*) = \left| \frac{x^*}{2(x^* - 2)} \right|.$$

This is not defined at $x^* = 2$. Hence the function $f(x)$ is numerically unstable and ill-conditioned for values of x close to 2. □

REMARK 1.9 If $x = f(x) = 0$, then the condition number is simply $|f'(x)|$. If $x = 0, f(x) \neq 0$ (or $f(x) = 0, x \neq 0$) then it is more useful to consider the relation between absolute errors than relative errors. The condition number then becomes $|f'(x)/f(x)|$. □

REMARK 1.10 Generally, if a numerical approximation $\tilde{z}$ to a quantity z is computed, the relative error is related to the number of digits after the decimal point that are correct. For example if $z = 0.0000123453$ and $\tilde{z} = 0.00001234543$, we say that $\tilde{z}$ is correct to 5 *significant digits*. Expressing z as 0.123453×10^{-4} and $\tilde{z}$ as 0.123454×10^{-4} , we see that if we round $\tilde{z}$ to the nearest number with five digits in its mantissa, all of those digits are correct, whereas, if we do the same with six digits, the sixth digit is not correct. Significant digits is the more logical way to talk about accuracy in a floating point computation where we are interested in relative error, rather than “number of digits after the decimal point,” which can have a different meaning. (Here, one might say that $\tilde{z}$ is correct to 9 digits after the decimal point.) □

1.2.2 Practicalities and the IEEE Floating Point Standard

Prior to 1985, different machines used different word lengths and different bases, and different machines rounded, chopped, or did something else to form the internal representation $fl(x)$ for real numbers x . For example, IBM mainframes generally used hexadecimal arithmetic ($\beta = 16$), with 8 hexadecimal digits total (for the base, sign, and exponent) in “single precision” numbers and 16 hexadecimal digits total in “double precision” numbers. Machines such as the Univac 1108 and Honeywell Multics systems used base $\beta = 2$ and 36

binary digits (or “bits”) total in single precision numbers and 72 binary digits total in double precision numbers. An unusual machine designed at Moscow State University from 1955-1965, the “Setun” even used base-3 ($\beta = 3$, or “ternary”) numbers. Some computers had 32 bits total in single precision numbers and 64 bits total in double precision numbers, while some “super-computers” (such as the Cray-1) had 64 bits total in single precision numbers and 128 bits total in double precision numbers.

Some hand-held calculators in existence today (such as some Texas Instruments calculators) can be viewed as implementing decimal (base 10, $\beta = 10$) arithmetic, say, with $L = -999$ and $U = 999$, and $t = 14$ digits in the mantissa.

Except for the Setun (the value of whose ternary digits corresponded to “positive,” “negative,” and “neutral” in circuit elements or switches), digital computers are mostly based on binary switches or circuit elements (that is, “on” or “off”), so the base β is usually 2 or a power of 2. For example, the IBM hexadecimal digit could be viewed as a group of 4 binary digits³.

Older floating point implementations did not even always fit exactly into the model we have previously described. For example, if x is a number in the system, then $-x$ may not have been a number in the system, or, if x were a number in the system, then $1/x$ may have been too large to be representable in the system.

To promote predictability, portability, reliability, and rigorous error bounding in floating point computations, the Institute of Electrical and Electronics Engineers (IEEE) and American National Standards Institute (ANSI) published a standard for binary floating point arithmetic in 1985: *IEEE/ANSI 754-1985: Standard for Binary Floating Point Arithmetic*, often referenced as “IEEE-754,” or simply “the IEEE standard⁴.” Almost all computers in existence today, including personal computers and workstations based on Intel, AMD, Motorola, etc. chips, implement most of the IEEE standard.

In this standard, $\beta = 2$, 32 bits total are used in a single precision number (an “IEEE single”), and 64 bits total are used for a double precision number (“IEEE double”). In a single precision number, 1 bit is used for the sign, 8 bits are used for the exponent, and $t = 23$ bits are used for the mantissa. In double precision numbers, 1 bit is used for the sign, 11 bits are used for the exponent, and 52 bits are used for the mantissa. Thus, for single precision numbers, the exponent is between 0 and $(1111111)_2 = 255$, and 128 is subtracted from this, to get an exponent between -127 and 128. In IEEE numbers, the minimum and maximum exponent are used to denote special symbols (such as infinity and “unnormalized” numbers), so the exponent in single precision represents magnitudes between $2^{-126} \approx 10^{-38}$ and $2^{127} \approx 10^{38}$. The

³An exception is in some systems for business calculations, where base 10 is implemented.

⁴An update to the 1985 standard was made in 2008. This update gives clarifications of certain ambiguous points, provides certain extensions, and specifies a standard for decimal arithmetic.

mantissa for single precision numbers represents numbers between $(2^0 = 1$ and $\sum_{i=0}^{23} 2^{-i} = 2(1 - 2^{-24}) \approx 2$. Similarly, the exponent for double precision numbers is, effectively, between $2^{-1022} \approx 10^{-308}$ and $2^{1023} \approx 10^{308}$, while the mantissa for double precision numbers represents numbers between $2^0 = 1$ and $\sum_{i=0}^{52} 2^{-i} \approx 2$.

Summarizing, the parameters for IEEE arithmetic appear in Table 1.1.

TABLE 1.1: Parameters for IEEE arithmetic

precision	β	L	U	t
single	2	-126	127	23
double	2	-1022	1023	52

In many numerical computations, such as solving the large linear systems arising from partial differential equation models, more digits or a larger exponent range is required than is available with IEEE single precision. For this reason, many numerical analysts at present have adopted IEEE double precision as the default precision. For example, underlying computations in the popular computational environment MATLAB are done in IEEE double precision.

IEEE arithmetic provides four ways of defining $f(x)$, that is, four “rounding modes,” namely, “round down,” “round up,” “round to nearest,” and “round to zero,” are specified as follows.

round down: $f(x) = x \downarrow$, the nearest machine number to the real number x that is less than or equal to x

round up: $f(x) = x \uparrow$, the nearest machine number to the real number x that is greater than or equal to x .

round to nearest: $f(x)$ is the nearest machine number to the real number x .

round to zero: $f(x)$ is the nearest machine number to the real number x that is closer to 0 than x . This corresponds to “chop” as explained on page 11.

The four elementary operations $+$, $-$, $\times$, and $/$ must be such that $f(x \odot y)$ is implemented for all four rounding modes, for $\odot \in \{-, +, \times, /, \sqrt{\cdot}\}$.

The default mode (if the rounding mode is not explicitly set) is normally “round to nearest,” to give an approximation after a long string of computations that is hopefully near the exact value. If the mode is set to “round down” and a string of computations is done, then the result is less than or

equal to the exact result. Similarly, if the mode is set to “round up,” then the result of a string of computations is greater than or equal to the exact result. In this way, mathematically rigorous bounds on an exact result can be obtained. (This technique must be used astutely, since naive use could result in bounds that are too large to be meaningful.)

Several parameters more directly related to numerical computations than L , U , and t are associated with any floating point number system. These are

HUGE: the largest representable number in the floating point system;

TINY: the smallest positive representable number in the floating point system.

ϵ_m : the *machine epsilon*, the smallest positive number which, when added to 1, gives something other than 1 when using the rounding mode—round to the nearest.

These so-called “machine constants” appear in Table 1.2 for the IEEE single and IEEE double precision number systems.

TABLE 1.2: Machine constants for IEEE arithmetic

Precision	HUGE	TINY	ϵ_m
single	$2^{127} \approx 3.40 \cdot 10^{38}$	$2^{-126} \approx 1.18 \cdot 10^{-38}$	$2^{-23} \approx 1.19 \cdot 10^{-7}$
double	$2^{1023} \approx 1.79 \cdot 10^{308}$	$2^{-1022} \approx 2.23 \cdot 10^{-308}$	$2^{-52} \approx 2.22 \cdot 10^{-16}$

For IEEE arithmetic, $1/\text{TINY} < \text{HUGE}$, but $1/\text{HUGE} < \text{TINY}$. This brings up the question of what happens when the result of a computation has absolute value less than the smallest number representable in the system, or has absolute value greater than the largest number representable in the system. In the first case, an *underflow* occurs, while, in the second case, an *overflow* occurs. In floating point computations, it is usually (but not always) reasonable to replace the result of an underflow by 0, but it is usually more problematical when an overflow occurs. Many systems prior to the IEEE standard replaced an underflow by 0 but stopped when an overflow occurred.

The IEEE standard specifies representations for special numbers ∞ , $-\infty$, $+\text{0}$, $-\text{0}$, and NaN , where the latter represents “not a number.” The standard specifies that computations do not stop when an overflow or underflow occurs, or when quantities such as $\sqrt{-1}$, $1/0$, $-1/0$, etc. are encountered (although many programming languages by default or optionally do stop). For example, the result of an overflow is set to ∞ , whereas the result of $\sqrt{-1}$ is set to NaN , and computation continues. The standard also specifies “gradual underflow,” that is, setting the result to a “denormalized” number, or a number in the

floating point format whose first digit in the mantissa is equal to 0. Computation rules for these special numbers, such as $\text{NaN} \times \text{any number} = \text{NaN}$, $\infty \times \text{any positive normalized number} = \infty$, allow such “nonstop” arithmetic.

Although the IEEE nonstop arithmetic is useful in many contexts, the numerical analyst should be aware of it and be cautious in interpreting results. In particular, algorithms may not behave as expected if many intermediate results contain ∞ or NaN , and the accuracy is less than expected when denormalized numbers are used. In fact, many programming languages, by default or with a controllable option, stop if ∞ or NaN occurs, but implement IEEE nonstop arithmetic with an option.

Example 1.16

(Illustration of underflow and overflow) Suppose, for the purposes of illustration, we have a system with $\beta = 10$, $t = 2$ and one digit in the exponent, so that the positive numbers in the system range from 0.10×10^{-9} to 0.99×10^9 , and suppose we wish to compute $N = \sqrt{x_1^2 + x_2^2}$, where $x_1 = x_2 = 10^6$. Then both x_1 and x_2 are exactly represented in the system, and the nearest floating point number in the system to N is 0.14×10^7 , well within range. However, $x_1^2 = 10^{12}$, larger than the maximum floating point number in the system. In older systems, an overflow usually would result in stopping the computation, while in IEEE arithmetic, the result would be assigned the symbol “Infinity.” The result of adding “Infinity” to “Infinity” then taking the square root would be “Infinity,” so that N would be assigned “Infinity.” Similarly, if $x_1 = x_2 = 10^{-6}$, then $x_1^2 = 10^{-12}$, smaller than the smallest representable machine number, causing an “underflow.” On older systems, the result is usually set to 0. On IEEE systems, if “gradual underflow” is switched on, the result either becomes a denormalized number, with less than full accuracy, or is set to 0; without gradual underflow on IEEE systems, the result is set to 0. When the result is set to 0, a value of 0 is stored in N , whereas the closest floating point number in the system is 0.14×10^{-5} , well within range. To avoid this type of catastrophic underflow and overflow in the computation of N , we may use the following scheme.

1. $s \leftarrow \max\{|x_1|, |x_2|\}$.
2. $\eta_1 \leftarrow x_1/s$; $\eta_2 \leftarrow x_2/s$.
3. $N \leftarrow s\sqrt{\eta_1^2 + \eta_2^2}$.

□

1.2.2.1 Input and Output

For examining the output to large numerical computations arising from mathematical models, plots, graphs, and movies comprised of such plots and graphs are often preferred over tables of values. However, to develop such

models and study numerical algorithms, it is necessary to examine individual numbers. Because humans are trained to comprehend decimal numbers more easily than binary numbers, the binary format used in the machine is usually converted to a decimal format for display or printing. In many programming languages and environments (such as all versions of Fortran, C, C++, and in MATLAB), the format is of a form similar to $\pm d_1.d_2d_3\dots d_m e \pm \delta_1\delta_2\delta_3$, or $\pm d_1.d_2d_3\dots d_mE \pm \delta_1\delta_2\delta_3$, where the “e” or “E” denotes the “exponent” of 10. For example, $-1.00e+003$ denotes $-1 \times 10^3 = -1000$. Numbers are usually also input either in a standard decimal form (such as 0.001) or in this exponential format (such as $1.0e-3$). (This notation originates from the earliest computers, where the only output was a printer, and the printer could only print numerical digits and the 26 upper case letters in the Roman alphabet.)

Thus, for input, a decimal fraction needs to be converted to a binary floating point number, while, for output, a binary floating point number needs to be converted to a decimal fraction. This conversion necessarily is inexact. For example, the exact decimal fraction 0.1 converts to the infinitely repeating binary expansion $(0.00011)_2$, which needs to be rounded into the binary floating point system. The IEEE 754 standard specifies that the result of a decimal to binary conversion, within a specified range of input formats, be the nearest floating point number to the exact result, over a specified range, and that, within a specified range of formats, a binary to decimal conversion be the nearest number in the specified format (which depends on the number m of decimal digits requested to be printed).

Thus, the number that one sees as output is usually not exactly the number that is represented in the computer. Furthermore, while the floating point operations on binary numbers are usually implemented in hardware or “firmware” independently of the software system, the decimal to binary and binary to decimal conversions are usually implemented separately as part of the programming language (such as Fortran, C, C++, Java, etc.) or software system (such as MATLAB). The individual standards for these languages, if there are any, may not specify accuracy for such conversions, and the languages sometimes do not conform to the IEEE standard. That is, the number that one sees printed may not even be the closest number in that format to the actual number.

This inexactness in conversion usually does not cause a problem, but may cause much confusion in certain instances. In those instances (such as in “debugging,” or finding programming blunders), one may need to examine the binary numbers directly. One way of doing this is in an “octal,” or base-8 format, in which each digit (between 0 and 7) is interpreted as a group of three binary digits, or in hexadecimal format (where the digits are 0-9, A, B, C, D, E, F), in which each digit corresponds to a group of four binary digits.

1.2.2.2 Standard Functions

To enable accurate computation of elementary functions such as $\sin$, $\cos$, and $\exp$, IEEE 754 specifies that a “long” 80-bit register (with “guard digits”) be available for intermediate computations. Furthermore, IEEE 754-2008, an official update to IEEE 754-1985, provides a list of functions it recommends be implemented, and specifies accuracy requirements (in terms of *correct rounding*), for those functions a programming language elects to implement.

REMARK 1.11 Alternative number systems, such as variable precision arithmetic, multiple precision arithmetic, rational arithmetic, and combinations of approximate and symbolic arithmetic have been investigated and implemented. These have various advantages over the traditional floating point arithmetic we have been discussing, but also have disadvantages, and usually require more time, more circuitry, or both. Eventually, with the advance of computer hardware and better understanding of these alternative systems, their use may become more ubiquitous. However, for the foreseeable future, traditional floating point number systems will be the primary tool in numerical computations. $\square$

1.3 Interval Computations

Interval computations are useful for two main purposes:

- to use floating point computations to compute mathematically rigorous bounds on an exact result (and hence to rigorously bound roundoff error);
- to use floating point computations to compute mathematically rigorous bounds on the ranges of functions over boxes.

In complicated traditional floating point algorithms, naive arrangement of interval computations usually gives bounds that are too wide to be of practical use. For this reason, interval computations have been ignored by many. However, used cleverly and where appropriate, interval computations are powerful, and provide rigor and validation when other techniques cannot.

Interval computations are based on interval arithmetic.

1.3.1 Interval Arithmetic

In interval arithmetic, we define operations on intervals, which can be considered as ordered pairs of real numbers. We can think of each interval as

representing the range of possible values of a quantity. The result of an operation is then an interval that represents the range of possible results of the operation as the range of all possible values, as the first argument ranges over all points in the first interval and the second argument ranges over all values in the second interval. To state this symbolically, let $\mathbf{x} = [\underline{x}, \bar{x}]$ and $\mathbf{y} = [\underline{y}, \bar{y}]$, and define the four elementary operations by

$$\mathbf{x} \odot \mathbf{y} = \{x \odot y \mid x \in \mathbf{x} \text{ and } y \in \mathbf{y}\} \text{ for } \odot \in \{+, -, \times, \div\}. \quad (1.6)$$

Interval arithmetic's usefulness derives from the fact that the mathematical characterization in Equation (1.6) is equivalent to the following *operational definitions*.

$$\left. \begin{aligned} \mathbf{x} + \mathbf{y} &= [\underline{x} + \underline{y}, \bar{x} + \bar{y}], \\ \mathbf{x} - \mathbf{y} &= [\underline{x} - \bar{y}, \bar{x} - \underline{y}], \\ \mathbf{x} \times \mathbf{y} &= [\min\{\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y}\}, \max\{\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y}\}] \\ \frac{1}{\mathbf{x}} &= [\frac{1}{\bar{x}}, \frac{1}{\underline{x}}] \quad \text{if } \underline{x} > 0 \text{ or } \bar{x} < 0 \\ \mathbf{x} \div \mathbf{y} &= \mathbf{x} \times \frac{1}{\mathbf{y}} \end{aligned} \right\} \quad (1.7)$$

The ranges of the four elementary interval arithmetic operations are exactly the ranges of the corresponding real operations, but, if such operations are composed, *bounds on the ranges* of real functions can be obtained. For example, if

$$f(x) = (x + 1)(x - 1), \quad (1.8)$$

then

$$f([-2, 2]) = ([-2, 2] + 1)([-2, 2] - 1) = [-1, 3][-3, 1] = [-9, 3],$$

which contains the exact range $[-1, 3]$.

REMARK 1.12 In some definitions of interval arithmetic, division by intervals containing 0 is defined, consistent with (1.6). For example,

$$\frac{[1, 2]}{[-3, 4]} = \left[-\infty, -\frac{1}{3}\right] \cup \left[\frac{1}{4}, \infty\right] = \mathbb{R}^* \setminus \left(-\frac{1}{3}, \frac{1}{4}\right),$$

where $\mathbb{R}^*$ is the *extended real number system*,⁵ consisting of the real numbers with the two additional numbers $-\infty$ and ∞ . This extended interval arith-

⁵also known as the *two-point compactification* of the real numbers

metic⁶ was originally invented by William Kahan⁷ for computations with continued fractions, but has wider use than that. Although a closed system can be defined for the sets arising from this extended arithmetic, typically, the complements of intervals (i.e., the unions of two semi-infinite intervals) are immediately intersected with intervals, to obtain zero, one, or two intervals. Interval arithmetic can then proceed using (1.7). $\square$

The power of interval arithmetic lies in its implementation on computers. In particular, *outwardly rounded* interval arithmetic allows *rigorous enclosures* for the ranges of operations and functions. This makes a *qualitative* difference in scientific computations, since the results are now intervals in which the exact result *must* lie. It also enables use of floating point computations for automated theorem proving.

Outward rounding can be implemented on any machine that has downward rounding and upward rounding, such as any machine that complies with the IEEE 754 standard. For example, take $\mathbf{x} + \mathbf{y} = [\underline{x} + \underline{y}, \overline{x} + \overline{y}]$. If $\underline{x} + \underline{y}$ is computed with downward rounding, and $\overline{x} + \overline{y}$ is computed with upward rounding, then the resulting interval $\mathbf{z} = [\underline{z}, \overline{z}]$ that is represented in the machine *must* contain the exact range of $x + y$ for $x \in \mathbf{x}$ and $y \in \mathbf{y}$. We call the expansion of the interval from rounding the lower end point down and the upper end point up *roundout error*.

Interval arithmetic is only *subdistributive*. That is, if $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ are intervals, then

$$\mathbf{x}(\mathbf{y} + \mathbf{z}) \subseteq \mathbf{x}\mathbf{y} + \mathbf{x}\mathbf{z}, \text{ but } \mathbf{x}(\mathbf{y} + \mathbf{z}) \neq \mathbf{x}\mathbf{y} + \mathbf{x}\mathbf{z} \text{ in general.} \quad (1.9)$$

As a result, algebraic expressions that would be equivalent if real values are substituted for the variables are not equivalent if interval values are used. For example, if, instead of writing $(x - 1)(x + 1)$ for $f(x)$ in (1.8), suppose we write

$$f(x) = x^2 - 1, \quad (1.10)$$

and suppose we provide a routine that computes an enclosure for the range of x^2 that is the exact range to within roundoff error. Such a routine could be as follows:

ALGORITHM 1.1

(Computing an interval whose end points are machine numbers and which encloses the range of x^2 .)

⁶There are small differences in current definitions of extended interval arithmetic. For example, in some systems, $-\infty$ and ∞ are not considered numbers, but just descriptive symbols. In those systems, $[1, 2]/[-3, 4] = (-\infty, -1/3] \cup [1/4, \infty) = \mathbb{R} \setminus (-1/3, 1/4)$. See [71] for a theoretical analysis of extended arithmetic.

⁷who also was a major contributor to the IEEE 754 standard

INPUT: $\mathbf{x} = [\underline{x}, \bar{x}]$.

OUTPUT: a machine-representable interval that contains the range of x^2 over $\mathbf{x}$.

IF $\underline{x} \geq 0$ THEN

RETURN $[\underline{x}^2, \bar{x}^2]$, where $\underline{x}^2$ is computed with downward rounding and $\bar{x}^2$ is computed with upward rounding.

ELSE IF $\bar{x} \leq 0$ THEN

RETURN $[\bar{x}^2, \underline{x}^2]$, where $\bar{x}^2$ is computed with downward rounding and $\underline{x}^2$ is computed with upward rounding.

ELSE

1. Compute $\underline{x}^2$ and $\bar{x}^2$ with both downward and upward rounding; that is, compute $\underline{x}_l^2$ and $\underline{x}_u^2$ such that $\underline{x}_l^2$ and $\underline{x}_u^2$ are machine representable numbers and $\underline{x}^2 \in [\underline{x}_l^2, \underline{x}_u^2]$, and compute $\bar{x}_l^2$ and $\bar{x}_u^2$ such that $\bar{x}_l^2$ and $\bar{x}_u^2$ are machine representable numbers and $\bar{x}^2 \in [\bar{x}_l^2, \bar{x}_u^2]$.
2. RETURN $[0, \max\{\underline{x}_u^2, \bar{x}_u^2\}]$.

END IF

END ALGORITHM 1.1.

With Algorithm 1.1 and rewriting $f(x)$ from (1.8) as in (1.10), we obtain

$$\mathbf{f}([-2, 2]) = [-2, 2]^2 - 1 = [0, 4] - 1 = [-1, 3]$$

which, in this case, is equal to the exact range of f over $[-2, 2]$.

In fact, this illustrates a general principle: If each variable in the expression occurs only once, then interval arithmetic gives the exact range, to within roundout error. We state this formally as

THEOREM 1.9

(Fundamental theorem of interval arithmetic.) Suppose $f(x_1, x_2, \dots, x_n)$ is an algebraic expression in the variables x_1 through x_n (or a computer program with inputs x_1 through x_n), and suppose that this expression is evaluated with interval arithmetic. The algebraic expression or computer program can contain the four elementary operations and operations such as x^n , $\sin(x)$, $\exp(x)$, and $\log(x)$, etc., as long as the interval values of these functions contain their range over the input intervals. Then

1. The interval value $\mathbf{f}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ contains the range of f over the interval vector (or box) $(\mathbf{x}_1, \dots, \mathbf{x}_n)$.

2. If the single functions (the elementary operations and functions x^n , etc.) have interval values that represent their exact ranges, and if each variable x_i , $1 \leq i \leq n$ occurs only once in the expression for f , then the values of f obtained by interval arithmetic represent the exact ranges of f over the input intervals.

If the expression for f contains one or more variables more than once, then overestimation of the range can occur due to *interval dependency*. For example, when we evaluate our example function $f([-2, 2])$ according to (1.8), the first factor, $[-1, 3]$ is the exact range of $x + 1$ for $x \in [-2, 2]$, while the second factor, $[-3, 1]$ is the exact range of $x - 1$ for $x \in [-2, 2]$. Thus, $[-9, 3]$ is the exact range of $\tilde{f}(x_1, x_2) = (x_1 + 1)(x_2 - 1)$ for x_1 and x_2 independent, $x_1 \in [-2, 2]$, $x_2 \in [-2, 2]$.

We now present some definitions and theorems to clarify the practical consequences of interval dependency.

DEFINITION 1.6 *An expression for $f(x_1, \dots, x_n)$ which is written so that each variable occurs only once is called a single use expression, or SUE.*

Fortunately, we do not need to transform every expression into a single use expression for interval computations to be of value. In particular, the interval dependency becomes less as the widths of the input intervals becomes smaller. The following formal definition will help us to describe this precisely.

DEFINITION 1.7 *Suppose an interval evaluation $\mathbf{f}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ gives $[a, b]$ as a result interval, but the exact range $\{f(x_1, \dots, x_n), x_i \in \mathbf{x}_i, 1 \leq i \leq n\}$ is $[c, d] \subseteq [a, b]$. We define the excess width $E(\mathbf{f}; \mathbf{x}_1, \dots, \mathbf{x}_n)$ in the interval evaluation $\mathbf{f}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ by $E(\mathbf{f}; \mathbf{x}_1, \dots, \mathbf{x}_n) = (c - a) + (b - d)$.*

For example, the excess width in evaluating $f(x)$ represented as $(x+1)(x-1)$ over $\mathbf{x} = [-2, 2]$ is $(-1 - (-9)) + (3 - 3) = 8$. In general, we have

THEOREM 1.10

Suppose $f(x_1, x_2, \dots, x_n)$ is an algebraic expression in the variables x_1 through x_n (or a computer program with inputs x_1 through x_n), and suppose that this expression is evaluated with interval arithmetic, as in Theorem 1.9, to obtain an interval enclosure $\mathbf{f}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ to the range of f for $x_i \in \mathbf{x}_i$, $1 \leq i \leq n$. Then, if $E(\mathbf{f}; \mathbf{x}_1, \dots, \mathbf{x}_n)$ is as in Definition 1.7, we have

$$E(\mathbf{f}; \mathbf{x}_1, \dots, \mathbf{x}_n) = \mathcal{O}\left(\max_{1 \leq i \leq n} w(\mathbf{x}_i)\right),$$

where $w(\mathbf{x})$ denotes the width of the interval $\mathbf{x}$.

That is, the overestimation becomes less as the uncertainty in the arguments to the function becomes smaller.

Interval evaluations as in Theorem 1.10 are termed *first-order* interval extensions. It is not difficult to obtain *second-order* extensions, where required. (See Exercise 26 below.)

1.3.2 Application of Interval Arithmetic: Examples

We give one such example here.

Example 1.17

Using 4-digit decimal floating point arithmetic, compute an interval enclosure for the first two digits of e , and prove that these two digits are correct. $\square$

Solution: The fifth degree Taylor polynomial representation for e is

$$e = 1 + 1 + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \frac{1}{5!} + \frac{1}{6!}e^\xi,$$

for some $\xi \in [0, 1]$. If we assume we know $e < 3$ and we assume we know e^x is an increasing function of x , then the error term is bounded by

$$\left| \frac{1}{6!}e^\xi \right| \leq \frac{3}{6!} < 0.005,$$

so this fifth-degree polynomial representation should be adequate. We will evaluate each term with interval arithmetic, and we will replace e^ξ with $[1, 3]$. We obtain the following computation:

$$\begin{aligned} [1.000, 1.000] + [1.000, 1.000] &\rightarrow [2.000, 2.000] \\ [1.000, 1.000]/[2.000, 2.000] &\rightarrow [0.5000, 0.5000] \\ [2.000, 2.000] + [0.5000, 0.5000] &\rightarrow [2.500, 2.500] \\ [1.000, 1.000]/[6.000, 6.000] &\rightarrow [0.1666, 0.1667] \\ [2.500, 2.500] + [0.1666, 0.1667] &\rightarrow [2.666, 2.667] \\ [1.000, 1.000]/[24.00, 24.00] &\rightarrow [0.04166, 0.04167] \\ [2.666, 2.667] + [0.04166, 0.04167] &\rightarrow [2.707, 2.709] \\ [1.000, 1.000]/[120.0, 120.0] &\rightarrow [0.008333, 0.008334] \\ [2.707, 2.709] + [0.008333, 0.008334] &\rightarrow [2.715, 2.718] \\ [1.000, 1.000]/[720.0, 720.0] &\rightarrow [0.001388, 0.001389] \\ [0.001388, 0.001389] \times [1, 3] &\rightarrow [0.001388, 0.004167] \\ [2.715, 2.718] + [0.001388, 0.004167] &\rightarrow [2.716, 2.723] \end{aligned}$$

Since we used outward rounding in these computations, this constitutes a mathematical proof that $e \in [2.716, 2.723]$.

Note:

1. These computations can be done automatically on a computer, as simply as evaluating the function in floating point arithmetic. We will explain some programming techniques for this in Chapter 6, Section 6.2.
2. The solution is illustrative. More sophisticated methods, such as argument reduction, would be used in practice to bound values of e^x more accurately and with less operations.

Proofs of the theorems, as well as greater detail, appear in various texts on interval arithmetic. A good book on interval arithmetic is R. E. Moore's classic text [58] although numerous more recent monographs and reviews are available. A World Wide Web search on the term "interval computations" will lead to some of these.

A general introduction to interval computations is [57]. That work gives not only a complete introduction, with numerous examples and explanation of pitfalls, but also provides examples with INTLAB, a free MATLAB toolbox for interval computations, and reference material for INTLAB. If you have MATLAB available, we recommend INTLAB for the exercises in this book involving interval computations.

1.4 Exercises

1. Answer the following:

- a) Prove that $|e^x - e^y| \leq |x - y|$, $\forall x, y \leq 0$.
- b) Show that

$$py^{p-1}(x - y) \leq x^p - y^p \leq px^{p-1}(x - y),$$

for $0 \leq y \leq x$, and $p \geq 1$.

- c) Assume $f(x)$ is continuous for $a \leq x \leq b$ and let

$$S = \sum_{j=1}^n w_j f(x_j),$$

with $x_j \in [a, b]$, $w_j \geq 0$ for $j = 1, \dots, n$, and $\sum_{j=1}^n w_j = 1$. Show that $S = f(\xi)$ for some $\xi \in [a, b]$.

- d) Let $f' \in C[a, b]$ and $f'(x) \neq 0$ for all x in (a, b) . Determine at how many points the function $f(x)$ can possibly vanish in $[a, b]$. Explain your answer.

2. Write down a polynomial $p(x)$ such that $|\text{sinc}(x) - p(x)| \leq 10^{-10}$ for $-0.2 \leq x \leq 0.2$, where

$$\text{sinc}(x) = \begin{cases} \frac{\sin(x)}{x} & \text{if } x \neq 0, \\ 1 & \text{if } x = 0 \end{cases}$$

is the “sinc” function (well-known in signal processing, etc.). Prove that your polynomial p satisfies the condition $|\text{sinc}(x) - p(x)| \leq 10^{-10}$ for $x \in [-0.2, 0.2]$.

Hint: You can obtain polynomial approximations with error terms for $\text{sinc}(x)$ by writing down Taylor polynomials and corresponding error terms for $\sin(x)$, then dividing these by x . This can be easier than trying to differentiate $\text{sinc}(x)$. For the proof part, you can use, for example, the Taylor polynomial remainder formula or the alternating series test and you can use interval arithmetic to obtain bounds.

3. Suppose f has a continuous third derivative. Show that

$$\left| \frac{f(x+h) - f(x-h)}{2h} - f'(x) \right| = \mathcal{O}(h^2).$$

4. Suppose f has a continuous fourth derivative. Show that

$$\left| \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} - f''(x) \right| = \mathcal{O}(h^2).$$

5. Let $x_n = e^{1/n} - 1/n - 1$ and $\alpha_n = 1/n$. Show that $x_n = \mathcal{O}(\alpha_n^2)$.

6. Let $x_n = \frac{n+1}{n^2 \ln n}$ and $\alpha_n = 1/n$. Show that $x_n = o(\alpha_n)$.

7. Let $a = 0.326$, $b = 0.000135$, and $c = 0.000431$. Assuming 3-digit decimal computer arithmetic with rounding to nearest, does $a + (b+c) = (a+b) + c$ when using this arithmetic?

8. Let $a = 0.41$, $b = 0.36$, and $c = 0.7$. Assuming a 2-digit decimal computer arithmetic with rounding, show that $\frac{a-b}{c} \neq \frac{a}{c} - \frac{b}{c}$ when using this arithmetic.

9. Let x_i^* for $i = 1, 2, 3, 4$ be positive numbers on a computer. With a unit round-off error δ , $x_i^* = x_i (1 + \epsilon_i)$ with $|\epsilon_i| \leq \delta$, where x_i for $i = 1, 2, 3, 4$ are the exact numbers. Consider the scalar product $S_2 = a^T b$ where $a = [x_1, x_2]^T$ and $b = [x_3, x_4]^T$. Let S_2^* be the floating point approximation of S_2 . Prove that $S_2^*/S_2 \leq e^{4\delta}$.

10. Suppose IEEE single precision with rounding to nearest is being used. What is the maximum relative error $|(x - fl(x))/x|$, for

- (a) $x \in [2^{-30}, 2^{-27}]$?
 - (b) $x \in [10000000, 10100000]$?
 - (c) $x \in [2^{60}, 2^{62}]$?
11. Repeat Exercise 10, but assume that IEEE double precision rather than IEEE single precision is used.
 12. Write down a formula relating the unit roundoff δ of Remark 1.5 (page 12) and the machine epsilon ϵ_m defined on page 20.
 13. Suppose, for illustration, we have a system with base $\beta = 10$, $t = 3$ decimal digits in the mantissa, and $L = -9$, $U = 9$ for the exponent. For example, 0.123×10^4 , that is, 1230 is a machine number in this system. Suppose also that “round to nearest” is used in this system.
 - (a) What is HUGE for this system?
 - (b) What is TINY for this system?
 - (c) What is the machine epsilon ϵ_m for this system?
 - (d) Let $f(x) = \sin(x) + 1$.
 - i. Write down $f(f(0))$ and $f(f(0.0008))$ in normalized format for this toy system.
 - ii. Compute $f(f(f(0.0008)) - f(f(0)))$. On the other hand, what is the nearest machine number to the exact value of $f(0.0008) - f(0)$?
 - iii. Compute $f(f(f(0.0008)) - f(f(0))) / f(0.0008)$. Compare this to the nearest machine number to the exact value of $(f(0.0008) - f(0)) / 0.0008$ and to $f'(0)$.
 14. Consider evaluation of $f(x) = \log(x + 1) - \log(x)$, for x large.
 - (a) Discuss the effects of roundoff error on the absolute error and relative error in the approximation of $f(x)$.
 - (b) Propose an alternate expression for evaluating $f(x)$ for large x that is less vulnerable to roundoff error. (Hint: The truncation error (“method error”) may be nonzero for such an expression, but could be negligible in relation to the unit roundoff error.)
 - (c) Test your predictions from part 14a and your expression from part 14b by making some computations for large x , say, using MATLAB (which employs double precision IEEE arithmetic), or, say, by writing a short Fortran, C, or C++ program.
 15. Let $f(x) = \frac{\ln(x+1) - \ln(x)}{2}$.
 - (a) Use four-digit decimal arithmetic with rounding to evaluate $f(1000)$.

- (b) Rewrite $f(x)$ in a form that avoids the loss of significant digits and evaluate $f(x)$ for $x = 1000$ once again.
- (c) Compare the relative errors for the answers obtained in (a) and (b).
16. Assume that x^* and y^* are approximations to x and y with relative errors r_x and r_y , respectively, and that $|r_x|, |r_y| < R$. Assume further that $x \neq y$. How small must R be in order to ensure that $x^* \neq y^*$?
17. Let x^* and y^* be the floating point representations of x and y , respectively. Let $f(x^*, y^*)$ be the approximate value of $f(x, y)$. Derive the relation between the relative error in evaluating the function $f(x, y)$ in terms of the relative error in evaluating x and the relative error in evaluating y .
18. The formula for the net capacitance when two capacitors of values x and y are connected in series is

$$z = \frac{xy}{x + y}.$$

Suppose the measured values of x and y are $x^* = 1$ and $y^* = 1$, respectively. Estimate the relative error in the function evaluation of

$$z^* = \frac{x^*y^*}{x^* + y^*}$$

given $|x - x^*| = 1$ and $|y - y^*| = 1$.

19. Show that the condition number of the product $f(x) \cdot g(x)$ of functions satisfies $\kappa_{fg}(x) \leq \kappa_f(x) + \kappa_g(x)$.
20. Compute the condition number of $f(x) = e^{\sqrt{x^2-1}}$, $x > 1$ and discuss any possible ill-conditioning.
21. Let $f(x) = x^{\frac{1}{10}}$ and let x^* approximate x correctly to k significant decimal digits (with rounding). Prove that $f(x^*)$ approximates $f(x)$ correctly to $(k + 1)$ significant decimal digits.
22. The function $f(x) = e^{x/2}$ is to be evaluated for any x , $(0 \leq x \leq 25)$ correct to 5 significant decimal digits. What digit decimal rounding arithmetic should be used (i.e., in x) to get the required accuracy in evaluating the function $f(x)$?
23. Define $I_n = \int_0^1 \frac{t^n}{t+1} dt$. Show that $I_0 = \ln(2)$ and that $I_n = \frac{1}{n} - I_{n-1}$ for $n \geq 1$. Describe the efficiency of computing I_j for $j = 1, \dots, 10$ using the recurrence formula obtained in part (a) and estimate the accuracy of I_{10} . Explain your observation.

24. Use the Taylor polynomial for $\arctan(x)$ centered at $x_0 = 0$ and interval arithmetic to compute mathematically rigorous bounds $\underline{\pi}$ and $\overline{\pi}$ on π , such that $\overline{\pi} - \underline{\pi} \leq 10^{-3}$.

Hint: You may use the interval arithmetic in the computer algebra systems Mathematica or Maple, or you may use the interval arithmetic in the INTLAB toolbox for MATLAB. An introduction to INTLAB, along with reference material for it, can be found in [57].

25. Let $f(x) = (\sin(x))^2 + x/2$. Use interval arithmetic to prove that there are no solutions to $f(x) = 0$ for $x \in [-1, -0.8]$.
26. Let $f(x) = x^2 - x$. One way of obtaining a bound on the range of f is by evaluating f directly, using interval arithmetic, while another way is by using the mean value theorem to obtain

$$\mathbf{f}_{\text{mv}}(\mathbf{x}) = \mathbf{f}(\tilde{x}) + \mathbf{f}'(\mathbf{x})(\mathbf{x} - \tilde{x}),$$

where $\tilde{x} \in \mathbf{x}$ is an approximation to the midpoint of $\mathbf{x}$ and where $\mathbf{f}'(\mathbf{x})$ is an interval evaluation of the derivative of f over $\mathbf{x}$. (This second bound on the range is usually called the *mean value extension* of f .)

- (a) For $\mathbf{x}_i = [1 - \epsilon_i, 1 + \epsilon_i]$, $\epsilon_i = 1/4^i$, $i = 1, 2, \dots, 10$, form a table whose columns are as follows:

i	4^{-i}	$\mathbf{x}_i$	$\mathbf{f}(\mathbf{x}_i)$	$w(\mathbf{f}(\mathbf{x}_i))$	$E(f; \mathbf{x})/w(\mathbf{x}_i)$	$E(f; \mathbf{x})/w(\mathbf{x}_i)^2$
-----	----------	----------------	----------------------------	-------------------------------	------------------------------------	--------------------------------------

where $E(f; \mathbf{x}) = w(\mathbf{f}(\mathbf{x}_i)) - w(\mathbf{f}^u(\mathbf{x}_i))$ denotes the “excess width” and where $\mathbf{f}^u(\mathbf{x})$ denotes the exact range of $\mathbf{f}$ over $\mathbf{x}$.

- (b) Do the same as in part 26a, except use $\mathbf{f}_{\text{mv}}(\mathbf{x})$ in place of $\mathbf{f}(\mathbf{x})$.
- (c) Based on your results in parts 26a and 26b, give values for α_1 and α_2 such that $E(f; \mathbf{x}) = \mathcal{O}(w(\mathbf{x})^{\alpha_1})$ and $E_{\text{mv}}(f; \mathbf{x}) = \mathcal{O}(w(\mathbf{x})^{\alpha_2})$. (In fact, your conclusions hold in general.)
27. Repeat Problem 26, but with intervals $\mathbf{x}_i = [2 - \epsilon_i, 2 + \epsilon_i]$.

Chapter 2

Numerical Solution of Nonlinear Equations of One Variable

2.1 Introduction

In this chapter, we study methods for finding approximate solutions to the scalar equation $f(x) = 0$. Some classical examples include the equation $x - \tan x = 0$ that occurs in the diffraction of light, or Kepler's equation $x - b \sin x = 0$ used for calculating planetary orbits. Other examples include transcendental equations such as $f(x) = e^x + x = 0$ and algebraic equations such as $x^7 + 4x^5 - 7x^2 + 6x + 3 = 0$. There are several reasons for beginning the study of numerical analysis with this problem:

1. The problem occurs frequently, i.e., the methods are useful.
 2. The problem illustrates the iterative method of solution, which is a common numerical technique.
 3. Convergence results are easy to derive.
 4. The problem illustrates that, when many methods are available, selection of the most suitable method depends on the particular problem.
-

2.2 Bisection Method

The bisection method is simple, reliable, and almost always can be applied, but is generally not as fast as other methods. Note that, if $y = f(x)$, then $f(x) = 0$ corresponds to the point where the curve $y = f(x)$ crosses the x -axis. The bisection method is based on the following result.

THEOREM 2.1

Suppose that $f \in C[a, b]$ and $f(a)f(b) < 0$. Then there is a $z \in [a, b]$ such that $f(z) = 0$.

PROOF The proof follows directly from the Intermediate Value Theorem. That is, if $f \in C[a, b]$ and k is any number between $f(a)$ and $f(b)$, then there is a $z \in [a, b]$ such that $f(z) = k$. Let $k = 0$. (See Figure 2.1.) $\square$

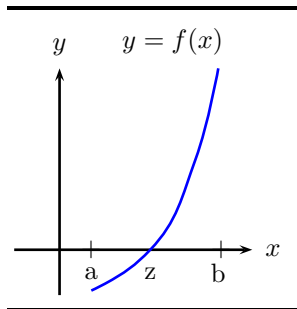


FIGURE 2.1: Example for a special case of the Intermediate Value Theorem (Theorem 2.1).

The *method of bisection* is simple to implement as illustrated in the following algorithm:

ALGORITHM 2.1

(The bisection algorithm)

INPUT: An error tolerance ϵ

OUTPUT: Either a point x that is within ϵ of a solution z or “failure to find a sign change”

1. Find a and b such that $f(a)f(b) < 0$. (By Theorem 2.1, there is a $z \in [a, b]$ such that $f(z) = 0$.) (Return with “failure to find a sign change” if such an interval cannot be found.)
2. Let $a_0 = a$, $b_0 = b$, $k = 0$.
3. Let $x_k = (a_k + b_k)/2$.
4. IF $f(x_k)f(a_k) > 0$ THEN
 - (a) $a_{k+1} \leftarrow x_k$,
 - (b) $b_{k+1} \leftarrow b_k$.
- ELSE
 - (a) $b_{k+1} \leftarrow x_k$,

(b) $a_{k+1} \leftarrow a_k$.

END IF

5. IF $(b_k - a_k)/2 < \epsilon$
THEN

Stop, since x_k is within ϵ of z . (See the explanation below.)

ELSE

(a) $k \leftarrow k + 1$.

(b) Return to step 3.

END IF

END ALGORITHM 2.1.

Basically, in the method of bisection, the interval $[a_k, b_k]$ contains z and $b_k - a_k = (b_{k-1} - a_{k-1})/2$. The interval containing z is reduced by a factor of 2 at each iteration.

Note: In practice, when programming bisection, we usually do not store the numbers a_k and b_k for all k as the iteration progresses. Instead, we usually store just two numbers a and b , replacing these by new values, as indicated in Step 4 of our bisection algorithm (Algorithm 2.1).

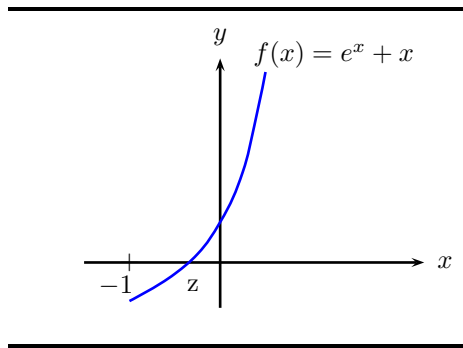


FIGURE 2.2: Graph of $e^x + x$ for Example 2.1.

Example 2.1

$f(x) = e^x + x$, $f(0) = 1$, $f(-1) = -0.632$. Thus, $-1 < z < 0$. (There is a unique zero, because $f'(x) = e^x + 1 > 0$ for all x .) Setting $a_0 = -1$ and $b_0 = 0$, we obtain the following table of values.

k	a_k	b_k	x_k
0	-1	0	$-1/2$
1	-1	$-1/2$	$-3/4$
2	$-3/4$	$-1/2$	-0.625
3	-0.625	-0.500	-0.5625
4	-0.625	-0.5625	-0.59375

Thus $z \in (-0.625, -0.5625)$; see Figure 2.2. $\square$

REMARK 2.1 The method always works for f continuous, as long as a and b can be found such that $f(a)f(b) < 0$ (and as long as we assume roundoff error does not cause us to incorrectly evaluate the sign of $f(x)$). However, consider $y = f(x)$ with $f(x) \geq 0$ for every x , but $f(z) = 0$. There are no a and b such that $f(a)f(b) < 0$. Thus, the method is not applicable to all problems in its present form. (See Figure 2.3 for an example of a root that cannot be found by bisection.) $\square$

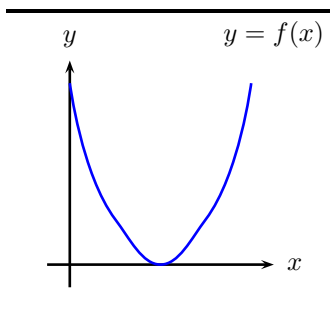


FIGURE 2.3: Example for Remark 2.1 (when f does not change sign in $[a, b]$).

We now have the question: How can we estimate the error $|x_k - z|$ in the k -th iteration?

THEOREM 2.2

Suppose that $f \in C[a, b]$ and $f(a)f(b) < 0$, then

$$|x_k - z| \leq \frac{b - a}{2^{k+1}}.$$

PROOF Combining for $k \geq 1$,

$$b_k - a_k = (b - a)/2^k,$$

$z \in (a_k, b_k)$, and $x_k = (a_k + b_k)/2$ gives

$$|z - x_k| \leq \frac{1}{2}(b_k - a_k) = (b - a)/2^{k+1}.$$

□

REMARK 2.2 Thus, in the algorithm, if $\frac{1}{2}(b_k - a_k) = (b - a)/2^{k+1} < \epsilon$, then $|z - x_k| < \epsilon$. □

Example 2.2

How many iterations are required to reduce the error to less than 10^{-6} if $a = 0$ and $b = 1$?

Solution: We need $\frac{1}{2^{k+1}}(1 - 0) < 10^{-6}$. Thus, $2^{k+1} > 10^6$, or $k = 19$. □

2.3 The Fixed Point Method

The method can be stated in terms of simple principles here. However, these principles are important both in the multidimensional analogue we describe in Section 8.2 (on page 442) and in infinite-dimensional analogues used in the mathematical analysis of differential equations. Let G be a closed subset of $\mathbb{R}$ or $\mathbb{C}$.

DEFINITION 2.1 $z \in G$ is a fixed point of g if $g(z) = z$.

REMARK 2.3 If $f(x) = g(x) - x$, then a fixed point of g is a zero of f . □

The *fixed-point iteration method* is defined by the following: For $x_0 \in G$,

$$x_{k+1} = g(x_k) \quad \text{for } k = 0, 1, 2, \dots$$

Example 2.3

$g(x) = \frac{1}{2}(x + 1)$, $x_0 = 0$, $x_{k+1} = \frac{1}{2}(x_k + 1)$. Then $x_0 = 0$, $x_1 = 1/2$, $x_2 = 3/4$, $x_3 = 7/8$, $x_4 = 15/16, \dots$ □

An important question is: when does $\{x_k\}_{k=0}^{\infty}$ converge to z , a fixed point of g ? Fixed-point iteration does not always converge. Consider $g(x) = x^2$, whose fixed points are $x = 0$ and $x = 1$. If $x_0 = 2$, then $x_{k+1} = x_k^2$, so $x_1 = 4$, $x_2 = 16$, $x_3 = 256, \dots$

The following definition is useful when discussing convergence of fixed point iteration.

DEFINITION 2.2 *g satisfies a Lipschitz condition on G if there is a Lipschitz constant $L \geq 0$ such that*

$$|g(x) - g(y)| \leq L|x - y| \text{ for all } x, y \in G. \quad (2.1)$$

If g satisfies (2.1) with $0 \leq L < 1$, g is said to be a *contraction* on the set G . We now have the following well-known result:

THEOREM 2.3

(Contraction Mapping Theorem in one variable) Suppose that g maps G into itself (i.e., if $x \in G$ then $g(x) \in G$) and g satisfies a Lipschitz condition with $0 \leq L < 1$ (i.e., g is a contraction on G). Then, there is a unique $z \in G$ such that $z = g(z)$, and the sequence determined by $x_0 \in G$, $x_{k+1} = g(x_k)$, $k = 0, 1, 2, \dots$ converges to z , with error estimates

$$|x_k - z| \leq \frac{L^k}{1 - L} |x_1 - x_0|, \quad k = 1, 2, \dots \quad (2.2)$$

$$|x_k - z| \leq \frac{L}{1 - L} |x_k - x_{k-1}|, \quad k = 1, 2, \dots \quad (2.3)$$

Before we prove the Contraction Mapping Theorem, we review the following concept.

DEFINITION 2.3 *A sequence $\{x_k\}_{k=1}^{\infty}$ is called a Cauchy sequence if, given any $\epsilon > 0$, there is an N dependent on ϵ such that $|x_k - x_\ell| < \epsilon$ for every k and ℓ greater than N . Cauchy sequences in $\mathbb{R}$ and $\mathbb{C}$ must converge to a point in $\mathbb{R}$ or $\mathbb{C}$, respectively. Number systems in which Cauchy sequences converge are called complete spaces.*

PROOF (of Theorem 2.3) Note that $x_k \in G$, $k = 0, 1, 2, \dots$, since $g(x) \in G$ if $x \in G$. Consider

$$\begin{aligned} |x_{k+1} - x_k| &= |g(x_k) - g(x_{k-1})| \\ &\leq L|x_k - x_{k-1}| \leq L^2|x_{k-1} - x_{k-2}| \leq \dots \leq L^k|x_1 - x_0|. \end{aligned}$$

Also,

$$\begin{aligned} |x_k - x_{k+j}| &\leq |x_k - x_{k+1}| + |x_{k+1} - x_{k+2}| + \dots + |x_{k+j-1} - x_{k+j}| \\ &\leq L(1 + L + \dots + L^{j-1})|x_k - x_{k-1}| \\ &\leq \frac{L}{1 - L} |x_k - x_{k-1}|. \end{aligned}$$

Hence,

$$|x_k - x_{k+j}| \leq \frac{L}{1-L} |x_k - x_{k-1}| \leq \frac{L^k}{1-L} |x_1 - x_0|. \quad (2.4)$$

Now, let $m = k$ and $n = k + j$. Then,

$$|x_m - x_n| \leq \frac{L^m}{1-L} |x_1 - x_0|. \quad (2.5)$$

Therefore, given $\epsilon > 0$, there is an N sufficiently large such that if $m, n > N$, $|x_n - x_m| < \epsilon$. (Recall that $0 \leq L < 1$.) Thus, $\{x_t\}_{t=0}^\infty$ is a Cauchy sequence, and converges, since $\mathbb{R}$ (or $\mathbb{C}$) is complete. However, since $x_{k+1} = g(x_k)$ and g is continuous,

$$z = \lim_{k \rightarrow \infty} x_{k+1} = \lim_{k \rightarrow \infty} g(x_k) = g(z).$$

Thus, $\{x_k\}$ converges to a fixed point of z .

Now consider uniqueness of z . Suppose that z_1 and z_2 are both fixed points, i.e., $z_1 = g(z_1)$ and $z_2 = g(z_2)$, and $z_1 \neq z_2$. Then

$$|z_2 - z_1| = |g(z_2) - g(z_1)| \leq L|z_2 - z_1| < |z_2 - z_1|.$$

This contradiction shows that the fixed point is unique. Note that to obtain (2.2) and (2.3), just let $j \rightarrow \infty$ in (2.4). $\square$

REMARK 2.4 Observe that $|x_k - z| = |g(x_{k-1}) - g(z)| \leq L|x_{k-1} - z|$. This inequality indicates that fixed-point iteration has at least a linear rate of convergence. $\square$

REMARK 2.5 There are two technical difficulties in applying the above theorem.

1. It may be difficult to find G such that g maps G into itself.
2. It may be difficult to show that g satisfies a Lipschitz condition on G .

$\square$

Throughout this section, we will show how to overcome these technical difficulties.

PROPOSITION 2.1

Suppose that g is continuously differentiable and that $|g'(x)| \leq L$ for $x \in G$. Then g satisfies a Lipschitz condition with Lipschitz constant L for $x \in G$.

PROOF By the Mean Value Theorem (Theorem 1.4 on page 3) followed by application of the assumption on $|g'(x)|$, we have

$$|g(y) - g(x)| = |g'(\xi)||y - x| \leq L|y - x|$$

for $x \in G$ and $y \in G$. □

Example 2.4

Suppose

$$g(x) = -\frac{x^3}{6} + \frac{x^5}{120},$$

and suppose we wish to find a Lipschitz constant for g over the interval $[-1/2, 1/2]$.

We will proceed by an interval evaluation of g' over $[-1/2, 1/2]$. Since $g'(x) = -x^2/2 + x^4/24$, we have

$$\begin{aligned} g'([-1/2, 1/2]) &\in -\frac{1}{2}[-1/2, 1/2]^2 + \frac{1}{24}[-1/2, 1/2]^4 \\ &= -\frac{1}{2}[0, 1/4] + \frac{1}{24}[0, 1/16] = [-1/8, 0] + [0, 1/384] \\ &\subseteq [-0.125, 0] + [0, 0.002605] \subseteq [-0.125, 0.00261]. \end{aligned}$$

Thus, since $|g'(x)| \leq \max_{y \in [-0.125, 0.00261]} |y| = 0.125$, g satisfies a Lipschitz condition with Lipschitz constant 0.125. □

Assume in the following that G is a closed subset of $\mathbb{R}$. Two useful results for showing that g satisfies the two conditions of the Contraction Mapping Theorem (Theorem 2.3 on page 40) are the following.

PROPOSITION 2.2

Suppose that g is continuously differentiable and $|g'(x)| \leq L < 1$ for $x \in G$, then g is a contraction on G .

PROOF Let x and $\tilde{x} \in G$. (Without loss of generality assume that $\tilde{x} > x$.) Then

$$|g(x) - g(\tilde{x})| = \left| \int_x^{\tilde{x}} g'(s) ds \right| \leq \int_x^{\tilde{x}} |g'(s)| ds \leq L|x - \tilde{x}|.$$

Thus, g is a contraction on G . □

PROPOSITION 2.3

Let $\rho > 0$ and $G = [c - \rho, c + \rho]$. Suppose that g is a contraction on G with Lipschitz constant L , $0 \leq L < 1$, and

$$|g(c) - c| \leq (1 - L)\rho.$$

Then g maps G into itself.

PROOF Suppose that $x \in G$. (We need to show that $g(x) \in G$.)

$$\begin{aligned} \text{Then } |g(x) - c| &\leq |g(x) - g(c) + g(c) - c| \leq |g(x) - g(c)| + |g(c) - c| \\ &\leq L|x - c| + (1 - L)\rho \leq L\rho + (1 - L)\rho = \rho. \end{aligned}$$

Thus, $g(x) \in G$. □

The following result is also useful.

PROPOSITION 2.4

Assume that z is a solution of $x = g(x)$, $g'(x)$ is continuous in an interval about z , and $|g'(z)| < 1$. Then g is a contraction in a sufficiently small interval about z , and g maps this interval into itself. Thus, provided x_0 is picked sufficiently close to z , the iterates will converge.

PROOF Select L such that $|g'(z)| < L < 1$. Then select an interval $I = [z - \epsilon, z + \epsilon]$ with $\max_{x \in I} |g'(x)| \leq L < 1$. We have that $g : I \rightarrow I$ since if $x \in I$,

$$|z - g(x)| = |g(z) - g(x)| = |g'(\xi)||z - x| \leq L|z - x| \leq \epsilon.$$

The contraction mapping theorem can then be used with $G = I$. □

Example 2.5

Let

$$g(x) = \frac{x}{2} + \frac{1}{x}.$$

Can we show that the fixed point iteration $x_{k+1} = g(x_k)$ converges for any starting point $x_0 \in [1, 2]$? We will use Proposition 2.2, Proposition 2.3, and Theorem 2.3 to show convergence. In particular, $g'(x) = 1/2 - 1/x^2$. Evaluating $g'(x)$ over $[1, 2]$ with interval arithmetic, we obtain

$$\begin{aligned} g'([1, 2]) &\in \frac{1}{2} - \frac{1}{[1, 2]^2} = \left[\frac{1}{2}, \frac{1}{2}\right] - \frac{1}{[1, 4]} \\ &= \left[\frac{1}{2}, \frac{1}{2}\right] - \left[\frac{1}{4}, 1\right] = \left[\frac{1}{2}, \frac{1}{2}\right] + \left[-1, -\frac{1}{4}\right] \\ &= \left[-\frac{1}{2}, \frac{1}{4}\right]. \end{aligned}$$

Thus, since $g'(x) \in g'([1, 2]) \in [-\frac{1}{2}, \frac{1}{4}]$ for every $x \in [1, 2]$,

$$|g'(x)| \leq \max_{x \in [-\frac{1}{2}, \frac{1}{4}]} |x| = \frac{1}{2}$$

for every $x \in [1, 2]$. Thus, from Proposition 2.2, g is a contraction on $[1, 2]$. Furthermore, letting $\rho = 1/2$ and $c = 3/2$, $|g(3/2) - 3/2| = 1/12 \leq 1/4$. Thus, by Proposition 2.3, g maps $[1, 2]$ into $[1, 2]$. Therefore, we can conclude from Theorem 2.3 that the fixed point iteration converges for any starting point $x_0 \in [1, 2]$ to the unique fixed point $z = g(z)$. $\square$

Example 2.6

Consider the fixed point equation

$$g(x) = b + \frac{a}{x}, \text{ with } a, b > 0.$$

Can we find z such that $z = g(z)$ and the fixed-point iterates converge to z ? The fixed-point iteration

$$x_{k+1} = g(x_k) = b + \frac{a}{x_k}$$

for this example gives

$$\begin{aligned} x_1 &= b + a/x_0 \\ x_2 &= b + a/x_1 = b + \frac{a}{b + \frac{a}{x_0}} \\ x_3 &= b + \frac{a}{x_2} = b + \frac{a}{b + \frac{a}{b + \frac{a}{x_0}}} \\ &\vdots \end{aligned}$$

Hence, we have the question whether this continued fraction expansion is converging to z , where $z = g(z)$. (Note that if $g(z) = z$, then $b + \frac{a}{z} = z$, $bz + a = z^2$, so

$$z = \frac{b + \sqrt{b^2 + 4a}}{2}$$

is the fixed point.) To apply the Contraction Mapping Theorem (Theorem 2.3 on page 40), we need to

- (a) determine a set G on which g is a contraction, and
- (b) make sure that G is such that $g : G \rightarrow G$.

For part (a), suppose that $G = [c - \rho, c + \rho]$. We have $g'(x) = -a/x^2$, so $|g'(x)| < 1$ for $x > \sqrt{a}$. Assume that $c, \rho > 0$. If $c - \rho > \sqrt{a}$, then $|g'(x)| < 1$. Let

$$\rho = \frac{c - \sqrt{a}}{2} \quad \text{and} \quad c = \frac{b + \sqrt{b^2 + 4a}}{2}.$$

Then $c - \rho > \sqrt{a}$, so g is a contraction on G .

For part (b), we need to show that g maps $[c - \rho, c + \rho]$ into itself. By Proposition 2.4, if $|g(c) - c| \leq (1 - L)\rho$ then $g : G \rightarrow G$. However, since $g(c) = c$, we need $0 \leq (1 - L)\rho$. But, $\rho = \frac{c - \sqrt{a}}{2} > 0$ and $0 < L < 1$, so $0 \leq (1 - L)\rho$.

For example, let $a = b = 1$. Then,

$$z = c = \frac{1 + \sqrt{5}}{2} \approx 1.62,$$

and we may take

$$G = [c - \rho, c + \rho] = [1.31, 1.93].$$

Let

$$x_0 = 1, \quad x_1 = 1 + \frac{1}{1} = 2, \quad x_2 = 1 + \frac{1}{1 + \frac{1}{1}} = 1.5 \in G, \dots$$

Thus, $x_3, x_4, x_5, \dots$ will be in G and will converge to $\frac{1 + \sqrt{5}}{2}$. That is,

$$1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1}}} \dots = \frac{1 + \sqrt{5}}{2}.$$

Note that $G \neq \mathbb{R}$ for this example. Consider $a = b = 1$, so $x_{k+1} = 1 + \frac{1}{x_k}$, and consider $x_0 = -\frac{1}{2}$, $x_1 = -1$. Then x_3 is undefined. $\square$

Example 2.7

Let $g(x) = 4 + \frac{1}{3} \sin 2x$ and $x_{k+1} = 4 + \frac{1}{3} \sin 2x_k$. Observing that

$$|g'(x)| = \left| \frac{2}{3} \cos 2x \right| \leq \frac{2}{3}$$

for all x shows that g is a contraction on all of $\mathbb{R}$, so we can take $G = \mathbb{R}$. Then $g : G \rightarrow G$ and g is a contraction on $\mathbb{R}$. Thus, for any $x_0 \in \mathbb{R}$, the iterations $x_{k+1} = g(x_k)$ will converge to z , where $z = 4 + \frac{1}{3} \sin 2z$. For $x_0 = 4$, the following values are obtained.

k	x_k
0	4
1	4.3298
2	4.2309
$\vdots$	$\vdots$
14	4.2615
15	4.2615

□

REMARK 2.6 Suppose that g satisfies the conditions of Theorem 2.3 (the Contraction Mapping Theorem) for $G = [a, b]$. Suppose also that $g(x) \leq g(y)$ for $a \leq x \leq y \leq b$, that is, g is a monotonically increasing function. (For example, g is monotonically increasing if $0 \leq g'(x) < 1$ for $x \in [a, b]$.) Let $x_0 = a$, so $x_1 = g(x_0) \geq x_0$. (We have $x_1 \geq a = x_0$ because $x_1 \in [a, b]$.) In fact, $x_2 = g(x_1) \geq g(x_0) = x_1$ since $x_1 \geq x_0$. Thus, $a = x_0 \leq x_1 \leq x_2 \leq \cdots$ and $x_k \rightarrow z$ monotonically as $k \rightarrow \infty$. Figure 2.4 illustrates this geometrically. A similar result holds if g is monotonically decreasing on $[a, b]$. □

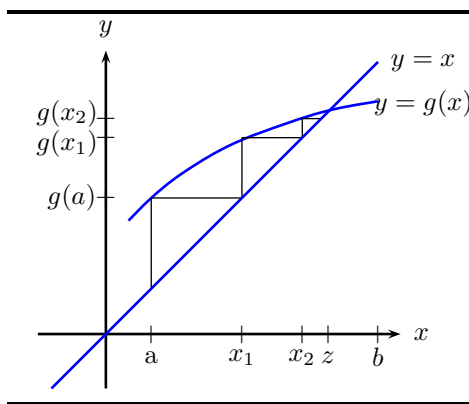


FIGURE 2.4: Example of Remark 2.6 (monotonic convergence of fixed point iteration).

We now consider an interesting result on the order of convergence for fixed-point iteration. Recall if $\lim_{k \rightarrow \infty} x_k = z$ and $|x_{k+1} - z| \leq c|x_k - z|^\alpha$, we say $\{x_k\}$ converges to z with rate of convergence α . (We specify that $c < 1$ for $\alpha = 1$.)

THEOREM 2.4

Assume that the iterations $x_{k+1} = g(x_k)$ converge to a fixed point z . Furthermore, assume that q is the first positive integer for which $g^{(q)}(z) \neq 0$ and if $q = 1$ then $|g'(z)| < 1$. Then the sequence $\{x_k\}$ converges to z with order q . (It is assumed that $g \in C^q(G)$, where G contains z .)

REMARK 2.7 Generally $q = 1$, so for many fixed-point iterations the order of convergence is linear. □

PROOF We have

$$\begin{aligned}
 |x_{k+1} - z| &= |g(x_k) - z| \\
 &= \left| g(z) + g'(z)(x_k - z) + \cdots + g^{(q)}(\xi_k) \frac{(x_k - z)^q}{q!} - z \right| \\
 &\quad \text{(where } \xi_k \text{ is between } x_k \text{ and } z) \\
 &= \frac{|g^{(q)}(\xi_k)|}{q!} |(x_k - z)^q|.
 \end{aligned}$$

Thus,

$$|x_{k+1} - z| \leq \max_{\xi \in G} \frac{|g^{(q)}(\xi)|}{q!} |x_k - z|^q = c |x_k - z|^q,$$

where

$$c = \max_{\xi \in G} \frac{|g^{(q)}(\xi)|}{q!}.$$

□

Example 2.8

Let

$$g(x) = \frac{x^2 + 6}{5}$$

and $G = [1, 2.3]$. Notice that $g : G \rightarrow G$ and

$$|g'(x)| = \left| \frac{2x}{5} \right| < 1$$

for $x \in G$. Then by Theorem 2.3 there is a unique fixed point $z \in G$. It is easy to see that the fixed point is $z = 2$. In addition, since $g'(z) = \frac{4}{5} \neq 0$, there is a linear rate of convergence. Inspecting the values in the following table, notice that the convergence is not fast.

k	x_k
0	2.2
1	2.168
2	2.140
3	2.116
4	2.095

□

Example 2.9

Let

$$g(x) = \frac{x}{2} + \frac{2}{x} = \frac{x^2 + 4}{2x}$$

be as in Example 2.5. It can be shown that if $0 < x_0 < 2$, then $x_1 > 2$. Also, $x_k > x_{k+1} > 2$ when $x_k > 2$. Thus, $\{x_k\}$ is a monotonically decreasing sequence bounded by 2 and hence is convergent. Thus, for any $x_0 \in (0, \infty)$, the sequence $x_{k+1} = g(x_k)$ converges to $z = 2$.

Now consider the convergence rate. We have that

$$g'(x) = \frac{1}{2} - \frac{2}{x^2},$$

so $g'(2) = 0$, and

$$g''(x) = \frac{4}{x^3},$$

so $g''(2) \neq 0$. By Theorem 2.4, the convergence is quadratic, and as indicated in the following table, the convergence is rapid.

k	x_k
0	2.2
1	2.00909
2	2.00002
3	2.00000000

□

Example 2.10

Let

$$g(x) = \frac{3}{8}x^4 - 4.$$

There is a unique fixed point $z = 2$. However, $g'(x) = \frac{3}{2}x^3$, so $g'(2) = 12$, and we cannot conclude linear convergence. Indeed, the fixed point iterations converge only if $x_0 = 2$. If $x_0 > 2$, then $x_1 > x_0 > 2$, $x_2 > x_1 > x_0 > 2, \dots$. Similarly, if $x_0 < 2$, it can be verified that, for some k , $x_k < 0$, after which $x_{k+1} > 2$, and we are in the same situation as if $x_0 > 2$. That is, fixed point iterations diverge unless $x_0 = 2$. □

Example 2.11

Consider the method given by $x_{k+1} = g(x_k)$ with

$$g(x) = x - \frac{f(x)}{f'(x)} - \frac{f''(x)}{2f'(x)} \left[\frac{f(x)}{f'(x)} \right]^2.$$

It is straightforward to show that if $f(z) = 0$, then $g'(z) = g''(z) = 0$ but $g'''(z) \neq 0$ (generally). Thus, when the method is convergent, the method has a cubic convergence rate. □

2.4 Newton's Method (Newton-Raphson Method)

We now return to the problem: given $f(x)$, find z such that $f(z) = 0$. Newton's iteration for finding approximate solutions to this problem has the form

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \quad \text{for } k = 0, 1, 2, \dots \quad (2.6)$$

REMARK 2.8 Newton's method is a special fixed-point method with $g(x) = x - f(x)/f'(x)$. $\square$

Figure 2.5 illustrates the geometric interpretation of Newton's method. To find x_{k+1} , the tangent line to the curve at point $(x_k, f(x_k))$ is followed to the x -axis. The tangent line is $y - f(x_k) = f'(x_k)(x - x_k)$. Thus, at $y = 0$, $x = x_k - f(x_k)/f'(x_k) = x_{k+1}$.

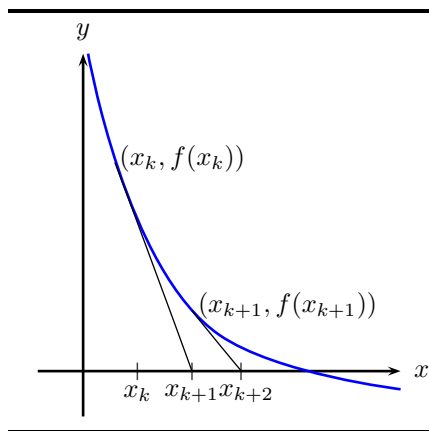


FIGURE 2.5: Illustration of two iterations of Newton's method.

REMARK 2.9 We will see that Newton's method is quadratically convergent, and is therefore fast when compared to a typical linearly convergent fixed point method. However, Newton's method may diverge if x_0 is not sufficiently close to a root z at which $f(z) = 0$. To see this, study Figure 2.6.

Another conceptually useful way of deriving Newton's method is using Taylor's formula. We have

$$0 = f(z) = f(x_k) + f'(x_k)(z - x_k) + \frac{(z - x_k)^2}{2} f''(\xi_k),$$

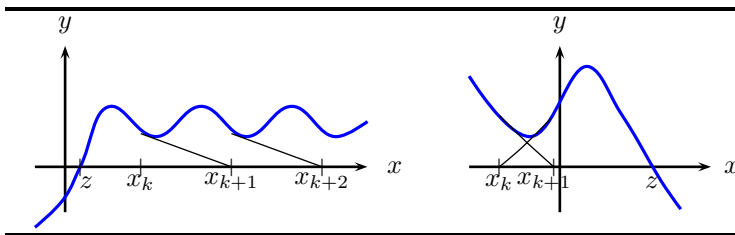


FIGURE 2.6: Example for Remark 2.9 (divergence of Newton's method). On the left, the sequence diverges; on the right, the sequence oscillates.

where ξ_k is between z and x_k . Thus, assuming that $(z - x_k)^2$ is small,

$$z \approx x_k - \frac{f(x_k)}{f'(x_k)}.$$

Hence, when $x_{k+1} = x_k - f(x_k)/f'(x_k)$, we would expect x_{k+1} to be closer to z than x_k . $\square$

We now study convergence of Newton's method. We make use of the following Lemma.

LEMMA 2.1

Let G be a subset of $\mathbb{R}$. Assume that $f \in C^2(G)$. Then for $x, y \in G$,

$$f(y) = f(x) + f'(x)(y - x) + R(y, x),$$

where

$$R(y, x) = (y - x)^2 \int_0^1 (1 - s) f''(x + s(y - x)) ds.$$

PROOF By Taylor's formula,

$$f(y) = f(x) + f'(x)(y - x) + \int_x^y (y - t) f''(t) dt.$$

Let $s = (t - x)/(y - x)$, $t = s(y - x) + x$, $dt = ds(y - x)$. Then

$$f(y) = f(x) + f'(x)(y - x) + \int_0^1 (y - x)^2 (1 - s) f''(x + s(y - x)) ds.$$

$\square$

THEOREM 2.5

(Convergence of Newton's method; existence and uniqueness of a root.) Let $G \subset \mathbb{R}$. Assume that $f \in C^2(G)$ and f satisfies

$$|f'(x)| \geq m, \quad |f''(x)| \leq M$$

for $x \in G$, where $m, M > 0$. Then for each zero $z \in G$, there is a neighborhood $K_\rho(z) = [z - \rho, z + \rho] \subseteq G$ such that z is the only zero of f in $K_\rho(z)$, and for each $x_0 \in K_\rho(z)$, the approximations $x_1, x_2, \dots$ remain in $K_\rho(z)$ and converge to z with error estimates:

$$(a) \quad |x_k - z| \leq \frac{2m}{M} q^{2^k}, \quad \text{where } q = \frac{M}{2m} |x_0 - z|,$$

$$(b) \quad |x_k - z| \leq \frac{1}{m} |f(x_k)| \leq \frac{M}{2m} |x_k - x_{k-1}|^2,$$

$$(c) \quad |x_{k+1} - z| \leq \frac{M}{2m} |x_k - z|^2 \quad (\text{quadratic convergence})$$

for $k = 1, 2, \dots$. Also, ρ can be selected to be any number less than $2m/M$ provided that $K_\rho(z) \subset G$.

Theorem 2.5 is a one-dimensional analog of the Newton–Kantorovich theorem, which we will see in §8.3.5 on page 454.

PROOF By the Mean Value Theorem,

$$\left| \frac{f(x) - f(\tilde{x})}{x - \tilde{x}} \right| = |f'(\xi)| \geq m$$

for some $\xi \in G$, whenever $x \in K_\rho(z)$ and $\tilde{x} \in K_\rho(z) \subseteq G$. Thus,

$$m|x - \tilde{x}| \leq |f(x) - f(\tilde{x})| \quad \text{for } x, \tilde{x} \in K_\rho(z) \subseteq G. \quad (2.7)$$

By Lemma 2.1,

$$f'(x)(x - y) = f(x) - f(y) + (y - x)^2 \int_0^1 (1 - s) f''(x + s(y - x)) ds.$$

Thus,

$$|f'(x)||x - y| \leq |f(x) - f(y)| + (y - x)^2 M \frac{1}{2} \quad (2.8)$$

Now choose $\rho < 2m/M$. Then there is a unique zero in $K_\rho(z)$. To see this, let $z_1 \neq z$ be another zero. Then by (2.8),

$$|f'(z_1)||z_1 - z| \leq \frac{1}{2} M |z_1 - z|^2.$$

Thus, $|z_1 - z| \geq 2m/M > \rho$, which is not possible since $K_\rho(z) = (z - \rho, z + \rho)$. Now consider $g(x) = x - f(x)/f'(x)$ for $x \in K_\rho(z)$. Then

$$g(x) - z = -\frac{1}{f'(x)} [f(x) + (z - x)f'(x)] = \frac{1}{f'(z)} R(z, x)$$

by Lemma 2.1. Thus,

$$|g(x) - z| \leq \frac{|R(z, x)|}{|f'(z)|} \leq \frac{1}{2} \frac{M}{m} |z - x|^2 \leq \rho^2 \frac{M}{2m} < \rho. \quad (2.9)$$

Hence, $g(x) \in [z - \rho, z + \rho] = K_\rho(z)$ for $x \in K_\rho(z)$. Thus, g maps $K_\rho(z)$ into $K_\rho(z)$. Let $x = x_k$, $x_{k+1} = g(x_k)$, $k = 1, 2, \dots$. Thus, the approximations remain in $K_\rho(z)$. Also, by (2.9) (letting $x = x_k$),

$$|x_{k+1} - z| \leq \frac{M}{2m} |x_k - z|^2$$

for $k = 0, 1, 2, \dots$ proving inequality (c). This inequality has the form

$$\frac{M}{2m} |x_{k+1} - z| \leq \left(\frac{M}{2m} \right)^2 |x_k - z|^2.$$

Now let $\rho_k = \frac{M}{2m} |x_k - z|$. Then

$$\rho_{k+1} \leq \rho_k^2 \leq (\rho_{k-1}^2)^2 \leq \dots, \quad \text{so } \rho_{k+1} \leq \rho_0^{2^k}.$$

Thus,

$$\frac{M}{2m} |x_k - z| \leq \left(\frac{M}{2m} |x_0 - z| \right)^{2^k}.$$

Hence,

$$|x_k - z| \leq \frac{2m}{M} q^{2^k}, \quad \text{where } q = \frac{M}{2m} |x_0 - z|.$$

Inequality (a) is thus proven. (Notice that since $\rho < \frac{2m}{M}$, the above inequality proves convergence for $x_0 \in K_\rho(z)$.)

Finally, to show (b), we have by (2.7) that

$$\begin{aligned} |x_k - z| &\leq \frac{1}{m} |f(x_k) - f(z)| = \frac{1}{m} |f(x_k)| \\ &= \frac{1}{m} |f(x_k) - f(x_{k-1}) - f'(x_{k-1})(x_k - x_{k-1})| \\ &= \frac{1}{m} |R(x_k, x_{k-1})| \quad \left(\text{since } x_k = x_{k-1} - \frac{f(x_{k-1})}{f'(x_{k-1})} \right) \\ &\leq \frac{M}{2m} |x_k - x_{k-1}|^2. \end{aligned}$$

□

REMARK 2.10 For $f(z) = 0$, $f'(z) \neq 0$, and $f \in C^2(G)$, there is some $K_\rho(z) \subset G$ such that Newton's iteration converges for $x_0 \in K_\rho(z)$. However, ρ may be quite small as indicated by Figure 2.6. □

REMARK 2.11 The quadratic convergence rate of Newton's method could have been inferred from Theorem 2.4 by analyzing Newton's method as a fixed point iteration. Consider

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} = g(x_k).$$

Observe that $g(z) = z$,

$$g'(z) = 0 = 1 - \frac{f'(z)}{f'(z)} + \frac{f(z)f''(z)}{(f'(z))^2},$$

and, usually, $g''(z) \neq 0$. Thus, the quadratic convergence follows from Theorem 2.4. $\square$

REMARK 2.12 By estimate (c), Newton's method has a quadratic convergence rate, i.e.,

$$|x_{k+1} - z| \leq \frac{M}{2m}|x_k - z|^2,$$

which is generally very fast. Consider estimate (a). If $q = 0.9$ and $k = 10$, then

$$|x_k - z| \leq \frac{2m}{M}(0.9)^{2^{10}} = \frac{2m}{M}(0.9)^{1024} = \frac{2m}{M} \times 1.39 \times 10^{-47}.$$

$\square$

Example 2.12

Let $f(x) = x + e^x$. Compare bisection, simple fixed-point iteration, and Newton's method.

- Newton's method: $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} = x_k - \frac{(x_k + e^{x_k})}{(1 + e^{x_k})} = \frac{(x_k - 1)e^{x_k}}{1 + e^{x_k}}.$
- Fixed-Point (one form): $x_{k+1} = -e^{x_k} = g(x_k).$

k	x_k (Bisection) $a = -1, b = 0$	x_k (Fixed-Point)	x_k (Newton's)
0	-0.5	-1.0	-1.0
1	-0.75	-0.367879	-0.537883
2	-0.625	-0.692201	-0.566987
3	-0.5625	-0.500474	-0.567143
4	-0.59375	-0.606244	-0.567143
5	-0.578125	-0.545396	-0.567143
10	-0.566895	-0.568429	-0.567143
20	-0.567143	-0.567148	-0.567143

$\square$

2.5 The Univariate Interval Newton Method

A simple application of the ideas behind Newton's method and the Mean Value Theorem leads to a mathematically rigorous computation of the zeros of a function f . In particular, suppose $\mathbf{x} = [\underline{x}, \bar{x}]$ is an interval, and suppose that there is a $z \in \mathbf{x}$ with $f(z) = 0$. Let $\tilde{x}$ be any point (such as the midpoint of $\mathbf{x}$). Then the Mean Value Theorem (page 3) gives

$$0 = f(\tilde{x}) + f'(\xi)(z - \tilde{x}). \quad (2.10)$$

Solving (2.10) for z , then applying the fundamental theorem of interval arithmetic (page 26) gives

$$\begin{aligned} z &= \tilde{x} - \frac{f(\tilde{x})}{f'(\xi)} \\ &\in \tilde{x} - \frac{f(\tilde{x})}{\mathbf{f}'(\mathbf{x})} = \mathbf{N}(f; \mathbf{x}, \tilde{x}). \end{aligned} \quad (2.11)$$

Thus, any solution $z \in \mathbf{x}$ of $f(x) = 0$ must also be in $\mathbf{N}(f; \mathbf{x}, \tilde{x})$.

DEFINITION 2.4 We call $\mathbf{N}(f; \mathbf{x}, \tilde{x})$ the univariate interval Newton operator.

The interval Newton operator forms the basis of a fixed-point type of iteration of the form

$$\mathbf{x}_{k+1} \leftarrow \mathbf{N}(f; \mathbf{x}_k, \tilde{x}_k) \text{ for } k = 1, 2, \dots$$

2.5.1 Existence and Uniqueness Verification

The interval Newton method is similar in many ways to the traditional Newton–Raphson method of Section 2.4 (page 49), but provides a way to use floating point arithmetic (with upward and downward roundings) to rigorously prove existence and uniqueness of solutions, as well as to provide rigorous upper and lower bounds on exact solutions. We now discuss existence and uniqueness properties of the interval Newton method. The first result is

THEOREM 2.6

Suppose $f \in C(\mathbf{x}) = C([\underline{x}, \bar{x}])$, $\tilde{x} \in \mathbf{x}$, and $\mathbf{N}(f; \mathbf{x}, \tilde{x}) \subseteq \mathbf{x}$. Then there is an $x^* \in \mathbf{x}$ such that $f(x^*) = 0$. Furthermore, this x^* is unique.

PROOF Assume that $\tilde{\mathbf{x}} = \mathbf{N}(f; \mathbf{x}, \tilde{x}) \subseteq \mathbf{x}$. First, observe that $0 \notin \mathbf{f}'(\mathbf{x})$, since $\mathbf{f}'(\mathbf{x})$ is in the denominator, and the only way $\mathbf{N}(f; \mathbf{x}, \tilde{x})$ can be a single

bounded interval is if the denominator does not contain zero. (See the remark on extended interval arithmetic on page 24.) Now, apply the Mean Value Theorem to obtain

$$f(\underline{x}) = f(\tilde{x}) + f'(\underline{\xi})(\underline{x} - \tilde{x}), \quad (2.12)$$

and to obtain

$$f(\overline{x}) = f(\tilde{x}) + f'(\overline{\xi})(\overline{x} - \tilde{x}), \quad (2.13)$$

for some $\underline{\xi} \in \underline{x}$ and some $\overline{\xi} \in \overline{x}$. Solving (2.12) for $f(\underline{x})/f'(\underline{\xi})$ and solving (2.13) for $f(\overline{x})/f'(\overline{\xi})$ gives

$$\frac{f(\underline{x})}{f'(\underline{\xi})} = \underline{x} - \left(\tilde{x} - \frac{f(\tilde{x})}{f'(\underline{\xi})} \right) \leq 0, \quad (2.14)$$

and

$$\frac{f(\overline{x})}{f'(\overline{\xi})} = \overline{x} - \left(\tilde{x} - \frac{f(\tilde{x})}{f'(\overline{\xi})} \right) \geq 0. \quad (2.15)$$

Then, because $0 \notin \mathbf{f}'(\mathbf{x})$, $\text{sgn}(f'(\underline{\xi})) = \text{sgn}(f'(\overline{\xi}))$, (2.14) and (2.15) imply that either

$$f(\underline{x}) \leq 0 \quad \text{and} \quad f(\overline{x}) \geq 0,$$

or

$$f(\underline{x}) \geq 0 \quad \text{and} \quad f(\overline{x}) \leq 0.$$

Therefore, by the Intermediate Value Theorem (Theorem 1.1 on page 1), there is an $x^* \in \hat{\mathbf{x}} \subseteq \mathbf{x}$ such that $f(x^*) = 0$.

To prove uniqueness of x^* , assume there is a $\hat{x} \in \mathbf{x}$ with $f(\hat{x}) = 0$ and $\hat{x} \neq x^*$. Then, by the Mean Value Theorem,

$$0 = f(\hat{x}) - f(x^*) = f'(\eta)(\hat{x} - x^*),$$

for some $\eta \in \mathbf{x}$, η between x^* and $\hat{x}$. Therefore, since $\hat{x} - x^* \neq 0$, $f'(\eta) = 0 \in \mathbf{f}'(\mathbf{x})$, contradicting our first deduction. Thus, x^* must be unique. $\square$

2.5.2 Interval Newton Algorithm

Now, let's study a formal algorithm for the interval Newton method.

ALGORITHM 2.2

(The univariate interval Newton method)

INPUT: $\mathbf{x} = [\underline{x}, \overline{x}]$, $f : \mathbf{x} \subset \mathbb{R} \rightarrow \mathbb{R}$, a maximum number of iterations N , and a stopping tolerance ϵ .

OUTPUT: Either

1. "solution does not exist within the original $\mathbf{x}$ ", or
2. a new interval $\mathbf{x}^*$ such that any $x^* \in \mathbf{x}$ with $f(x^*) = 0$ has $x^* \in \mathbf{x}^*$, and one of:

- (a) “existence and uniqueness verified” and “tolerance met.”
- (b) “existence and uniqueness not verified,”
- (c) “solution does not exist,” or
- (d) “existence and uniqueness verified” but “tolerance not met.”

1. $k \leftarrow 1$.

2. “existence and uniqueness verified” $\leftarrow$ “false.”

3. “solution does not exist” $\leftarrow$ “false.”

4. DO WHILE $k \leq N$.

(a) $\tilde{x} \leftarrow (\underline{x} + \overline{x})/2$.

(b) IF $\tilde{x} \notin \mathbf{x}$ THEN RETURN.

(c) $\tilde{\mathbf{x}} \leftarrow \mathbf{N}(f; \mathbf{x}, \tilde{x})$.

(d) IF $\tilde{\mathbf{x}} \subseteq \mathbf{x}$ (that is, if $\underline{\tilde{x}} \geq \underline{x}$ and $\overline{\tilde{x}} \leq \overline{x}$) THEN
“existence and uniqueness verified” $\leftarrow$ “true.”

(e) IF $\tilde{\mathbf{x}} \cap \mathbf{x} = \emptyset$ (that is, if $\overline{\tilde{x}} \leq \underline{x}$ or $\underline{\tilde{x}} \geq \overline{x}$) THEN
i. “solution does not exist” $\leftarrow$ “true.”
ii. RETURN.

(f) IF $w(\tilde{x}) < \epsilon$ THEN

i. $x^* \leftarrow \tilde{x}$.

ii. tolerance met $\leftarrow$ “true.”

iii. RETURN.

END IF

(g) $\mathbf{x} \leftarrow \mathbf{x} \cap \tilde{\mathbf{x}}$.

(h) $k \leftarrow k + 1$.

END DO

5. “tolerance met” $\leftarrow$ “false.”

6. RETURN.

END ALGORITHM 2.2.

Notes:

1. The interval Newton method generally becomes stationary. (That is, the end points of $\mathbf{x}$ can be proven to not change, under certain assumptions on the machine arithmetic.) However, it is good general programming practice to enforce an upper limit on the total number of iterations of any iterative process, to avoid problems arising from slow convergence, etc.

2. In Step 4a of Algorithm 2.2, the midpoint is computed approximately, and it occasionally occurs (when the interval is very narrow), that the machine approximation lies outside the interval. Thus, we need to check for this possibility.
3. Although f is evaluated at a point in the expression

$$\tilde{x} - \frac{f(\tilde{x})}{f'(x)}$$

for $N(f; \mathbf{x}, \tilde{x})$, the machine must evaluate f with interval arithmetic to take account of rounding error. (That is, we start with the computations with the degenerate interval $[\tilde{x}, \tilde{x}]$.) Otherwise, the results are not mathematically rigorous.

2.5.3 Convergence Rate of the Interval Newton Method

Similar to the traditional Newton–Raphson method, the interval Newton method exhibits quadratic convergence. (This is common knowledge.) An example of a specific theorem along these lines is

THEOREM 2.7

(Quadratic convergence of the interval Newton method) Suppose $f : \mathbf{x} \rightarrow \mathbb{R}$, suppose $f \in C(\mathbf{x})$ and $f' \in C(\mathbf{x})$, and suppose there is an $x^* \in \mathbf{x}$ such that $f(x^*) = 0$. Suppose further that f' is a first order or higher order interval extension of f in the sense of Theorem 1.10 (on page 27). Then, for the initial width $w(\mathbf{x})$ sufficiently small,

$$w(N(f; \mathbf{x}, \tilde{x})) = O(w(\mathbf{x}))^2.$$

We will not give a proof of Theorem 2.7 here, although Theorem 2.7 is a special case of Theorem 6.3, page 222 in [44]. We will illustrate this quadratic convergence with

Example 2.13

(Taken from [48].) Apply the interval Newton method $\mathbf{x} \leftarrow N(f; \mathbf{x}, \tilde{x})$, $\tilde{x} \leftarrow fl((\underline{x} + \overline{x})/2)$, to

$$f(x) = x^2 - 2,$$

starting with $\mathbf{x} = [1, 2]$ and $\tilde{x} = 1.5$. The results for Example 2.13 appear in Table 2.1. Here,

$$\delta_k = \frac{w(\mathbf{x}_k)}{\max \{ \max_{y \in \mathbf{x}_k} \{|y|\}, 1 \}}$$

is a scaled version of the width $w(\mathbf{x}_k)$, and $\rho_k = \max_{y \in f(\mathbf{x}_k)} \{|y|\}$. The displayed decimal intervals have been rounded out from the corresponding binary intervals. $\square$

TABLE 2.1: Convergence of the interval Newton method with $f(x) = x^2 - 2$.

k	$\mathbf{x}_k$	δ_k	ρ_k
0	[1.00000000000000, 2.00000000000000]	5.00×10^{-1}	2.00×10^0
1	[<u>1.37499999999999</u> , <u>1.43750000000001</u>]	4.35×10^{-2}	1.09×10^{-1}
2	[<u>1.41406249999999</u> , <u>1.41441761363637</u>]	2.51×10^{-4}	5.77×10^{-4}
3	[<u>1.41421355929452</u> , <u>1.41421356594718</u>]	4.70×10^{-9}	1.01×10^{-8}
4	[<u>1.41421356237309</u> , <u>1.41421356237310</u>]	4.71×10^{-16}	1.33×10^{-15}
5	[<u>1.41421356237309</u> , <u>1.41421356237310</u>]	4.71×10^{-16}	1.33×10^{-15}
6	[<u>1.41421356237309</u> , <u>1.41421356237310</u>]	4.71×10^{-16}	1.33×10^{-15}

2.6 Secant Method and Müller’s Method

Under certain circumstances, f may have a continuous derivative, but it may not be possible to explicitly compute it. This is less true now than in the past, because techniques of automatic differentiation (or “computational differentiation”), such as we explain in Section 6.2, page 327, have been developed, have become more widely available, and are used in practice. However, there are still various situations involving *black box* functions f . In “black box” functions, f is evaluated by some external procedure (such as a software system provided by someone other than its user), in which one supplies the input x , and the output $f(x)$ is returned, but the user (or the designer of the method for finding points x^* , $f(x^*) = 0$) does not have access to the internal workings, so that f' cannot be easily computed. In such cases, methods that converge more rapidly than the method of bisection, but that do not require evaluation of f' , are useful.

Example 2.14

Suppose we wish to find a zero of

$$f(x) = e^{-ax} - g\left(\cos x + \frac{a}{x} \sin x + \ln x\right),$$

where

$$g(x) = \frac{1}{1+x^2} + 3x^2 + 5x + h(5+e^x+\cos x),$$
$$h(x) = \frac{e^{2x^2}}{(1+x+x^2)},$$

and a is a constant. □

Problems as complicated as this are not uncommon. Prior to widespread use of automatic differentiation, applying Newton’s method to this problem

was quite difficult because it would have been difficult and time-consuming to calculate $f'(x_k)$ at each time step. Automatic differentiation is now an option for many problems of this type. However, in certain situations, such as applying the shooting method to solution of boundary-value problems (see the discussion in Chapter 10), f' cannot be directly computed and the secant method is useful. In this section, we will assume that f' cannot be computed, and we will treat f as a “black-box” function.

2.6.1 The Secant Method

In the secant method, $f'(x_k)$ is approximated by

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

The *secant method* thus has the form

$$x_{k+1} = x_k - f(x_k) \left[\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} \right]. \quad (2.16)$$

If $f(x_k)$ and $f(x_{k+1})$ have opposite signs, then, as with the bisection method, there must be an x^* between x_k and x_{k+1} for which $f(x^*) = 0$.

For the secant method, we need starting values x_0 and x_1 . However, only one evaluation of the function f is required at each iteration, since $f(x_{k-1})$ is known from the previous iteration.

Geometrically, (see figure 2.7), to obtain x_{k+1} , the secant to the curve through $(x_{k-1}, f(x_{k-1}))$ and $(x_k, f(x_k))$ is followed to the x-axis.

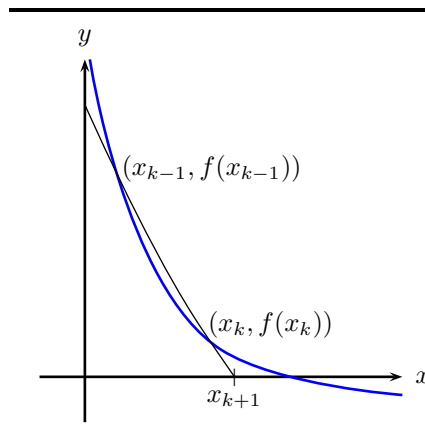


FIGURE 2.7: Geometric interpretation of the secant method.

REMARK 2.13 The secant method is not a fixed-point method, since $x_{k+1} = g(x_k, x_{k-1})$. $\square$

We now consider convergence of the secant method.

THEOREM 2.8

(Convergence of the secant method) Let G be a subset of $\mathbb{R}$ containing a zero z of $f(x)$. Assume $f \in C^2(G)$ and there exists an $M \geq 0$ such that

$$M = \frac{\max_{x \in G} |f''(x)|}{2 \min_{x \in G} |f'(x)|}.$$

Let x_0 and x_1 be two initial guesses to z and let

$$K_\epsilon(z) = (z - \epsilon, z + \epsilon) \in G,$$

where $\epsilon = \frac{\delta}{M}$ and $\delta < 1$. Let $x_0, x_1 \in K_\epsilon(z)$. Then, the iterates $x_2, x_3, x_4, \dots$ remain in $K_\epsilon(z)$ and converge to z with error

$$|x_k - z| \leq \frac{1}{M} \delta^{\left(\frac{1+\sqrt{5}}{2}\right)^k}.$$

REMARK 2.14 $(1 + \sqrt{5})/2 \approx 1.618$, a fractional order of convergence between 1 and 2. For Newton's method $|x_k - z| \leq q^{2^k}$ with $q < 1$. Compare this with the preceding bound. $\square$

PROOF (of Theorem 2.8) Subtracting z from both sides of the secant method and multiplying by -1 ,

$$z - x_{k+1} = z - x_k + f(x_k) \left[\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} \right].$$

This can be written as:

$$(z - x_{k+1}) = [-(z - x_{k-1})(z - x_k)] \left[\frac{C_k}{D_k} \right], \text{ where (since } f(z) = 0)$$

$$C_k = \left[\frac{f(z) - f(x_k)}{z - x_k} - \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \right] / (z - x_{k-1})$$

$$\text{and } D_k = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Now, let $e_k = z - x_k$. Then from the above equation,

$$|e_{k+1}| \leq |e_{k-1}| |e_k| \left| \frac{C_k}{D_k} \right|.$$

However, the Mean Value Theorem gives

$$D_k = f'(\xi_k)$$

for some ξ_k in the interval containing x_{k-1} and x_k . Similarly,

$$C_k = f''(\zeta_k)/2$$

for some ζ_k in the interval containing x_k , x_{k-1} , and z . To see this, suppose that $z > x_k > x_{k-1}$ (although true in general). Then Taylor's theorem gives

$$f(z) = f(x_k) + f'(x_k)(z - x_k) + f''(\hat{\zeta}_k) \frac{(z - x_k)^2}{2}$$

and

$$f(x_{k-1}) = f(x_k) + f'(x_k)(x_{k-1} - x_k) + f''(\hat{\zeta}_k) \frac{(x_{k-1} - x_k)^2}{2}.$$

Substituting into C_k ,

$$C_k = \frac{f''(\hat{\zeta}_k)}{2} \left(\frac{z - x_k}{z - x_{k-1}} \right) + \frac{f''(\hat{\zeta}_k)}{2} \left(\frac{x_k - x_{k-1}}{z - x_{k-1}} \right) = \frac{f''(\zeta_k)}{2}$$

by the Intermediate Value Theorem, since

$$\frac{z - x_k}{z - x_{k-1}} + \frac{x_k - x_{k-1}}{z - x_{k-1}} = 1.$$

Also, ζ_k is in the interval containing z , x_{k-1} , and x_k , since $\hat{\zeta}_k$ is between z and x_k and $\hat{\zeta}_k$ is between x_{k-1} and x_k . Thus,

$$|e_{k+1}| \leq |e_{k-1}| |e_k| \left| \frac{f''(\zeta_k)}{2f'(\xi_k)} \right|, \quad (2.17)$$

for some ζ_k and ξ_k in the interval containing z , x_k , and x_{k-1} . When $k = 1$, we have

$$|e_2| \leq |e_1| |e_0| M.$$

Thus,

$$M|e_2| \leq M|e_0|M|e_1| \leq \delta^2 \quad \text{if } x_0, x_1 \in K_\epsilon(z).$$

Also, $|e_2| \leq \epsilon^2 M = \epsilon \frac{\delta}{M} M \leq \epsilon$, so $x_2 \in K_\epsilon(z)$ if $x_0, x_1 \in K_\epsilon(z)$. Thus,

$$M|e_2| \leq M|e_0|M|e_1| \quad \text{and} \quad |e_2| \leq \epsilon.$$

Consider now

$$|e_3| \leq M|e_2| |e_1| \leq \epsilon$$

and

$$M|e_3| \leq M|e_2|M|e_1| \leq \delta^3.$$

Thus, in the k -th step, if $|e_k| \leq \epsilon$ and $M|e_k| \leq \delta^{q_k}$, then

$$|e_{k+1}| \leq \epsilon$$

and

$$M|e_{k+1}| \leq M|e_k|M|e_{k-1}| \leq \delta^{q_k+q_{k-1}} = \delta^{q_{k+1}}.$$

Hence,

$$\begin{cases} q_{k+1} = q_k + q_{k-1}, \\ q_0 = q_1 = 1, \end{cases} \quad (2.18)$$

that is, $\{q_k\}_{k=0}^\infty$ is the Fibonacci sequence. To solve the difference equation (2.18), we can assume the form $q_k = \alpha^k$. We then obtain

$$\alpha^{k+1} = \alpha^k + \alpha^{k-1},$$

so $\alpha^2 = \alpha + 1$. Solving this quadratic equation gives

$$\alpha = \frac{1 \pm \sqrt{5}}{2},$$

so

$$q_k = c_0 \left(\frac{1 + \sqrt{5}}{2} \right)^k + c_1 \left(\frac{1 - \sqrt{5}}{2} \right)^k \quad \text{with } q_0 = q_1 = 1.$$

Hence,

$$q_k = \frac{(1 + \sqrt{5})}{2\sqrt{5}} \left(\frac{1 + \sqrt{5}}{2} \right)^k - \frac{(1 - \sqrt{5})}{2\sqrt{5}} \left(\frac{1 - \sqrt{5}}{2} \right)^k.$$

Therefore,

$$q_k \leq \left(\frac{1 + \sqrt{5}}{2} \right)^k \quad (2.19)$$

for $k = 1, 2, 3, \dots$. Thus,

$$M|e_k| \leq \delta^{\left(\frac{1+\sqrt{5}}{2}\right)^k}$$

for $k = 1, 2, 3, \dots$, so $x_k \rightarrow z$ as $k \rightarrow \infty$. □

2.6.2 Müller's Method

We now consider a variant of the secant method called Müller's Method.

REMARK 2.15 Müller's method can be used to obtain real and complex roots of a function and thus is often used to find zeros of polynomials. $\square$

To define Müller's method, it is useful to use divided differences. (There is more on divided differences in Section 4.3.)

DEFINITION 2.5 We define

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

to be the first-order divided difference and

$$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}$$

to be the second-order divided difference.

REMARK 2.16 It can be shown that $f[x_0, x_1] = f'(\xi)$ for some ξ between x_0 and x_1 and $f[x_0, x_1, x_2] = \frac{1}{2}f''(\zeta)$ for some ζ between x_0 , x_1 , and x_2 . $\square$

We first present a geometric interpretation of Müller's method. Suppose that x_k , x_{k-1} , and x_{k-2} are approximations to z . To find x_{k+1} , we follow the quadratic polynomial through the points $(x_k, f(x_k))$, $(x_{k-1}, f(x_{k-1}))$, and $(x_{k-2}, f(x_{k-2}))$ to the x -axis. Let $p(x)$ be this quadratic polynomial. Then,

$$p(x) = f(x_k) + (x - x_k)f[x_k, x_{k-1}] + (x - x_k)(x - x_{k-1})f[x_k, x_{k-1}, x_{k-2}]. \quad (2.20)$$

(See §4.3.4.) Thus, setting $x = x_{k+1}$,

$$0 = f(x_k) + (x_{k+1} - x_k)f[x_k, x_{k-1}] + (x_{k+1} - x_k)(x_{k+1} - x_{k-1})f[x_k, x_{k-1}, x_{k-2}]. \quad (2.21)$$

Solving for x_{k+1} gives us Müller's method. (See Figure 2.8.) That is, we find the root of the quadratic through x_{k-2} , x_{k-1} , and x_k to obtain x_{k+1} . Rearranging to reduce rounding errors due to subtracting of nearly equal quantities, we obtain

$$x_{k+1} = x_k - \frac{2\lambda_k}{1 + \sqrt{1 - 4\mu_k\lambda_k}} \quad (2.22)$$

for $k = 2, 3, \dots$, where

$$\lambda_k = \frac{f(x_k)}{w_k}, \quad \mu_k = \frac{f[x_k, x_{k-1}, x_{k-2}]}{w_k},$$

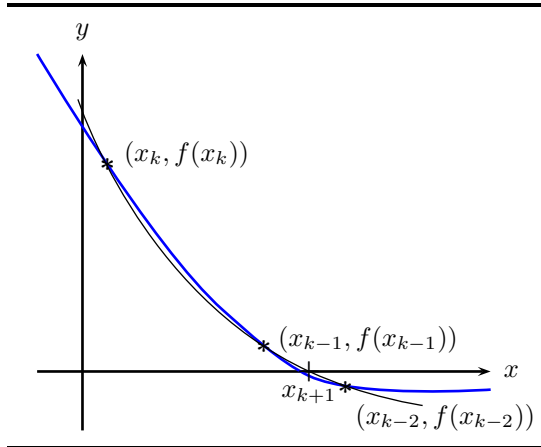


FIGURE 2.8: Geometric interpretation of Müller's method.

and

$$w_k = f[x_k, x_{k-1}] + (x_k - x_{k-1})f[x_k, x_{k-1}, x_{k-2}].$$

REMARK 2.17 The sign has been chosen in the quadratic formula for (2.22) so that x_{k+1} is the nearest solution to x_k of the quadratic equation (2.21). $\square$

REMARK 2.18 It can be shown that the order of convergence of Müller's method is about 1.83, which is a little faster than the secant method but slower than Newton's method. $\square$

REMARK 2.19 If $(1 - 4\mu_k\lambda_k) < 0$, then x_{k+1} is complex and the method, if convergent, converges to complex roots. However, even if all the roots are real, complex arithmetic must be used, because $1 - 4\mu_k\lambda_k$ may be less than zero at some iteration k . $\square$

2.7 Aitken Acceleration and Steffensen's Method

Linearly convergent sequences are obtained in many numerical methods, as we have seen in examples of fixed point iteration. In this section, we study a method useful for accelerating the convergence of linearly convergent sequences.

2.7.1 Aitken Acceleration

In this derivation, assume that $\{x_k\}$ is a linearly convergent sequence with limit α . That is,

$$\lim_{k \rightarrow \infty} \frac{x_{k+1} - \alpha}{x_k - \alpha} = \lambda \quad \text{where } -1 < \lambda < 1.$$

For example, for the fixed-point method with continuous g' , $\lambda = g'(\alpha)$, since

$$\alpha - x_{k+1} = g(x) - g(x_k) = g'(\xi_k)(\alpha - x_k),$$

for some ξ_k between α and x_k . Assume now that k is sufficiently large to ensure that $(x_{k+1} - \alpha) \approx \lambda(x_k - \alpha)$. Then

$$(x_{k+1} - \alpha) \approx \lambda(x_k - \alpha) \quad (2.23)$$

and, replacing k by $k+1$,

$$(x_{k+2} - \alpha) \approx \lambda(x_{k+1} - \alpha). \quad (2.24)$$

Solving (2.23) and (2.24) for α yields

$$\begin{aligned} \alpha &\approx x_k - \frac{(x_{k+1} - x_k)^2}{(x_{k+2} - x_{k+1}) - (x_{k+1} - x_k)} \\ &= x_{k+2} - \frac{(x_{k+2} - x_{k+1})^2}{(x_{k+2} - x_{k+1}) - (x_{k+1} - x_k)}. \end{aligned} \quad (2.25)$$

It is therefore reasonable to define another sequence $\{\hat{x}_k\}$ by

$$\hat{x}_k = x_k - \frac{(x_{k+1} - x_k)^2}{(x_{k+2} - x_{k+1}) - (x_{k+1} - x_k)} \quad \text{for } k = 0, 1, 2, \dots \quad (2.26)$$

We would expect $\{\hat{x}_k\}$ to converge to α faster than $\{x_k\}$. This procedure is called *Aitken acceleration* or extrapolation.

Example 2.15

Consider the fixed-point iteration

$$x_{k+1} = g(x_k) = \frac{x_k^2 + 6}{5}, \quad k = 0, 1, 2, \dots$$

In this problem, $\alpha = 2$.

k	x_k	$\hat{x}_k$
0	0	1.57895
1	1.2	1.82284
2	1.488	1.90213
3	1.64283	1.94259
4	1.73978	1.96516
5	1.80536	1.97853
6	1.85187	
7	1.88588	

In this example, Aitken acceleration clearly improves the convergence. $\square$

Example 2.16

This example illustrates that Aitken acceleration is not limited to fixed-point iteration. Consider

$$\int_0^1 f(x)dx \approx \sum_{i=1}^{2^k} h_k f(x_i) \quad \text{where } h_k = \frac{1}{2^k}, \quad x_i = ih_k - \frac{h_k}{2}$$

for $i = 1, 2, \dots, 2^k$, for integers $k \geq 0$. This method is the composite mid-point rule for approximating the integral, and can be shown to be linearly convergent. Let $f(x) = x^{-1/2}$ and

$$S_k = \sum_{i=1}^{2^k} h_k f(x_i).$$

The values obtained for this example are given in the following table:

k	S_k	$\widehat{S}_k$
1	1.57735	2.01314
2	1.69884	2.00325
3	1.78646	
4	1.84886	

Notice that $\alpha = \int_0^1 x^{-1/2}dx = 2$, so Aitken acceleration is effective in this example. $\square$

We have the following convergence result for Aitken acceleration.

THEOREM 2.9

Let $\{x_k\}$ be a linearly convergent sequence with limit α , such that the quantities $d_k = x_k - \alpha$ satisfy $d_{k+1} = (\lambda + \epsilon_k)d_k$, where λ is constant, $|\lambda| < 1$ and $\epsilon_k \rightarrow 0$ as $k \rightarrow \infty$. Then the sequence $\{\hat{x}_k\}$ derived from $\{x_k\}$ by (2.26) converges to α faster than $\{x_k\}$. That is,

$$\frac{\hat{x}_k - \alpha}{x_k - \alpha} \rightarrow 0 \text{ as } k \rightarrow \infty. \quad (2.27)$$

PROOF We have

$$d_{k+2} = (\lambda + \epsilon_{k+1})d_{k+1} = (\lambda + \epsilon_{k+1})(\lambda + \epsilon_k)d_k.$$

Hence,

$$\begin{aligned}(x_{k+2} - x_{k+1}) - (x_{k+1} - x_k) &= (d_{k+2} - d_{k+1}) - (d_{k+1} - d_k) \\ &= [(\lambda - 1)^2 + \epsilon'_k] d_k,\end{aligned}$$

where

$$\epsilon'_k = \lambda(\epsilon_k + \epsilon_{k+1}) - 2\epsilon_k + \epsilon_k \epsilon_{k+1}.$$

Since $\epsilon_k \rightarrow 0$, it is clear that $\epsilon'_k \rightarrow 0$ as $k \rightarrow \infty$. Thus, for k sufficiently large, say $k > k_0$, it follows that

$$(\lambda - 1)^2 + \epsilon'_k \neq 0,$$

so

$$(x_{k+2} - x_{k+1}) - (x_{k+1} - x_k) \neq 0,$$

so $\{\hat{x}_k\}$ is defined for $k > k_0$. We also have

$$x_{k+1} - x_k = d_{k+1} - d_k = (\lambda + \epsilon_k - 1)d_k.$$

Subtracting α from both sides of (2.26),

$$\begin{aligned}\hat{x}_k - \alpha &= d_k - \frac{(x_{k+1} - x_k)^2}{(x_{k+2} - x_{k+1}) - (x_{k+1} - x_k)} = d_k - \frac{(\lambda - 1 + \epsilon_k)^2 d_k}{(\lambda - 1)^2 + \epsilon'_k} \\ &= \left[\frac{\epsilon'_k - 2\epsilon_k(\lambda - 1) - \epsilon_k^2}{(\lambda - 1)^2 + \epsilon'_k} \right] d_k,\end{aligned}$$

recalling that $d_k = x_k - \alpha$. Hence,

$$\frac{\hat{x}_k - \alpha}{x_k - \alpha} = \frac{\epsilon'_k - 2\epsilon_k(\lambda - 1) - \epsilon_k^2}{(\lambda - 1)^2 + \epsilon'_k} \rightarrow 0 \text{ as } k \rightarrow \infty.$$

□

A relevant question is: Since all sequences are not linearly convergent, how do we determine when to apply Aitken acceleration? A measure for determining when to use Aitken acceleration is discussed in [41]. First, it can be shown that $\frac{d_3}{d_2} = \frac{d_2}{d_1} = \frac{d_1}{d_0} = \lambda$, i.e., the errors are linearly convergent, if and only if the points (x_0, x_1) , (x_1, x_2) , (x_2, x_3) lie on a straight line. Therefore, how closely a line fits these points indicates how well the errors are linearly converging. The serial correlation coefficient, a measure of the linearity of the three points, is given by

$$r = \frac{s_1 - s_2 s_3 / 3}{\sqrt{(s_4 - s_2^2 / 3)(s_5 - s_3^2 / 3)}},$$

where

$$s_1 = \sum_{i=1}^3 x_{i-1}x_i, \quad s_2 = \sum_{i=1}^3 x_{i-1}, \quad s_3 = \sum_{i=1}^3 x_i, \quad s_4 = \sum_{i=1}^3 x_{i-1}^2, \quad \text{and} \\ s_5 = \sum_{i=1}^3 x_i^2.$$

If r is close to ± 1 , e.g., $|r| > 0.999$, then Aitken acceleration is most effective. For Example 2.16, $r \approx 0.99998$.

2.7.2 Steffensen's Method

Aitken acceleration applied specifically to fixed-point iteration gives rise to a procedure known as Steffensen's method, which can be shown to be quadratically convergent under certain conditions. Consider $f(x) = 0$ and $x_{k+1} = f(x_k) + x_k$. (That is, $g(x) = x + f(x)$.) Then, Aitken acceleration gives

$$\begin{aligned} \hat{x}_k &= x_k - \frac{(x_{k+1} - x_k)^2}{(x_{k+2} - x_{k+1}) - (x_{k+1} - x_k)} \\ &= x_k - \frac{(f(x_k))^2}{f(x_{k+1}) - f(x_k)} = x_k - \frac{(f(x_k))^2}{f(f(x_k) + x_k) - f(x_k)} \end{aligned}$$

Letting $x_k = \hat{x}_{k-1}$ on the left hand side gives

$$\hat{x}_k = \hat{x}_{k-1} - \frac{(f(\hat{x}_{k-1}))^2}{f(f(\hat{x}_{k-1}) + \hat{x}_{k-1}) - f(\hat{x}_{k-1})}, \quad (2.28)$$

which is *Steffensen's method*.

REMARK 2.20 Steffensen's method is a fixed-point method

$$\hat{x}_k = S(\hat{x}_{k-1})$$

with

$$S(x) = x - \frac{(f(x))^2}{f(f(x) + x) - f(x)}.$$

□

REMARK 2.21 Steffensen's method applied to a fixed-point iteration $x_{k+1} = g(x_k)$ has the form $x_{k+1} = H(x_k)$, with

$$H(x_k) = x_k + \frac{(g(x_k) - x_k)^2}{(g(x_k) - x_k) - (g(g(x_k)) - g(x_k))}.$$

□

REMARK 2.22 By Remark 2.20, if $f(\alpha) = 0$, then $S(\alpha) = \alpha$ and $S'(\alpha) = 0$. Thus, by Theorem 2.4, page 46, if Steffensen's method is convergent, then it can converge quadratically. (You will show this in Exercise 16.) □

Example 2.17

Comparison of Newton's method and Steffensen's method for $f(x) = x^2 - 4$. (Note that $\alpha = 2$.)

$$\text{Newton's method: } x_{k+1} = x_k - \frac{x_k^2 - 4}{2x_k}.$$

Steffensen's method:

$$x_{k+1} = x_k - \frac{(x_k^2 - 4)^2}{(x_k^2 - 4 + x_k)^2 - 4 - x_k^2 + 4} = x_k - \frac{(x_k - 2)(x_k + 2)}{x_k^2 + 2x_k - 4}.$$

k	Newton's	Steffensen's
0	3	3
1	2.16667	2.545
2	2.00641	2.218
3	2.0000102	2.0463
4	2.0000000	2.00253
5	2.0000000	2.000008

(We see quadratic convergence for both methods, although Newton's method is faster.) □

2.8 Roots of Polynomials

In this section, we consider the special case of finding the roots of

$$f(x) = p(x) = a_0 + a_1x + \cdots + a_nx^n,$$

which is an important but difficult problem. We first review some important results on roots of polynomials.

THEOREM 2.10

(Fundamental theorem of algebra) If $p(x)$ is a polynomial of degree $n \geq 1$, then $p(x) = 0$ has at least one root (possibly complex).

COROLLARY 2.1

If

$$p(x) = a_0 + a_1x + \cdots + a_nx^n,$$

then there are n roots $z_1, z_2, \dots, z_n \in \mathbb{C}$ (which may be repeated), and

$$p(x) = a_n(x - z_1)(x - z_2) \cdots (x - z_n).$$

THEOREM 2.11

(Descartes' rule of signs) Let p be a polynomial with real coefficients, let ν_1 be the number of changes in the sign of the coefficients of $p(x)$, and let ν_2 be the number of changes in the sign of the coefficients of $p(-x)$. Let k_1 and k_2 be the number of positive real roots and negative real roots of $p(x)$. Then $k_1 \leq \nu_1$ and $\nu_1 - k_1$ is even, $k_2 \leq \nu_2$ and $\nu_2 - k_2$ is even.

Example 2.18

Consider $p(x) = x^r - x - a$, with $r > 1$ and $a > 0$. By Corollary 2.1, there are r real and complex roots of $p(x)$. Consider $p(-x) = (-1)^r x^r + x - a$. Clearly,

$$\nu_1 = 1 \text{ and } \nu_2 = \begin{cases} 1, & r \text{ even} \\ 2, & r \text{ odd.} \end{cases}$$

Thus, $k_1 = 1$, since $\nu_1 - k_1$ must be even; $k_2 = 1$ if $\nu_2 = 1$ and $k_2 = 2$ or 0 if $\nu_2 = 2$. Therefore, if r is even, there are one positive root and one negative root. If r is odd, then there are one positive root and two or zero negative roots. $\square$

We now have the following useful results on upper and lower bounds on the roots of a polynomial [8].

Result 1: $\max_k |z_k| \leq 1 + \max_{0 \leq i \leq n-1} \left| \frac{a_i}{a_n} \right| = R$. (All roots lie in a disc in the complex plane of radius R ; see Figure 2.9.)

Result 2: $\rho_1 \leq |z_k| \leq \rho_2$ for all k , where ρ_1 is the unique positive root of

$$|a_n|x^n + |a_{n-1}|x^{n-1} + \cdots + |a_1|x - |a_0| = 0$$

and ρ_2 is the unique positive root of

$$|a_n|x^n - |a_{n-1}|x^{n-1} - \cdots - |a_1|x - |a_0| = 0.$$

(All roots lie in an annulus in the complex plane; see Figure 2.10.)

REMARK 2.23 An efficient way to computationally evaluate a polynomial is using nested multiplication.

$$p(x) = a_0 + x(a_1 + x(a_2 + x(a_3 + \cdots + x(a_{n-1} + a_nx) \cdots)))$$

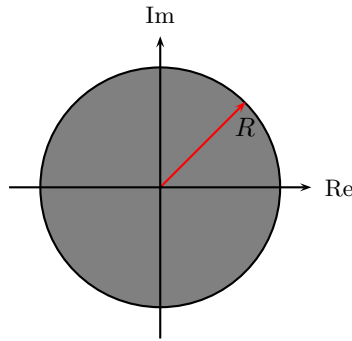


FIGURE 2.9: Disc in which roots must lie.

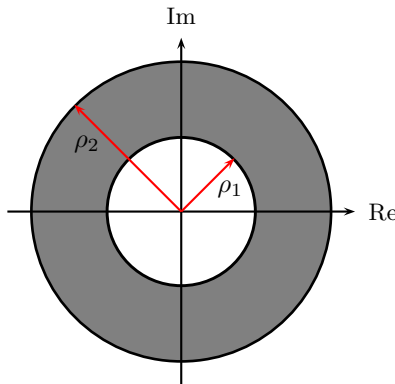


FIGURE 2.10: Annulus in which roots must lie.

Using nested multiplication, there are n multiplications and n additions in comparison with $2n - 1$ multiplications and n additions if $p(x)$ is evaluated directly.¹ $\square$

The following elementary but useful result relates nested multiplication to division of a polynomial by a linear factor.

THEOREM 2.12

(Horner's method or synthetic division) Let

$$p(x) = a_0 + a_1x + \cdots + a_nx^n.$$

Let $b_n = a_n$ and $b_k = a_k + b_{k+1}z$ for $k = n - 1, n - 2, \dots, 0$. Then $b_0 = p(z)$. Moreover, if

$$q(x) = b_1 + b_2x + \cdots + b_nx^{n-1},$$

¹This assumes that, in "direct evaluation," x^k is evaluated as $x \cdot x^{k-1}$ for $k = 2, \dots, n$.

then

$$p(x) = (x - z)q(x) + b_0.$$

PROOF We have

$$\begin{aligned} b_0 + (x - z)q(x) &= (x - z)(b_1 + b_2x + \cdots + b_nx^{n-1}) + b_0 \\ &= b_1x + b_2x^2 + \cdots + b_nx^n - (b_1z + b_2zx + \cdots + b_nzx^{n-1}) + b_0 \\ &= b_nx^n + (b_{n-1} - b_nz)x^{n-1} + (b_{n-2} - b_{n-1}z)x^{n-2} + \cdots + (b_0 - b_1z) \\ &= a_nx^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x + a_0. \end{aligned}$$

Thus, $p(x) = b_0 + (x - z)q(x)$ and $b_0 = p(z)$. □

Now consider a general procedure for finding the roots of a polynomial $p(x)$.

ALGORITHM 2.3

(Deflation method for finding all roots of a polynomial)

INPUT: the coefficients $\{a_i\}_{i=0}^n$ of the polynomial $p(x) = \sum_{k=0}^n a_k x^k$.

OUTPUT: approximations to the roots of the polynomial p .

1. Find root z_1 using some procedure such as Müller's method or Newton's method.²
2. Deflate $p(x) = (x - z_1)q(x)$ using Horner's method.
3. Find root z_2 of $q(x)$ using the numerical method in Step 1. Note that $q(x)$ is of degree $n - 1$.
4. Continue deflating and calculating roots.
5. After finding each root, iterate on that root, applying the method in Step 1, using the original polynomial $p(x)$.

END ALGORITHM 2.3.

REMARK 2.24 The roots may be sensitive to small changes in the coefficients. Thus, the errors introduced by rounding and approximation of z_1 , for example, can lead to large errors later. To help correct this problem, step (5) is added.

²Here, the root may be complex, so we will use complex arithmetic, and modify our geometric view of Newton's method or Müller's method.

To understand the sensitivity of the roots to perturbations in the coefficients, let's perform a stability analysis. Consider

$$p(x) = a_0 + a_1x + \cdots + a_nx^n$$

and

$$q(x) = b_0 + b_1x + \cdots + b_nx^n,$$

where $q(x)$ is a perturbation to $p(x)$. Let

$$p(x, \epsilon) = p(x) + \epsilon q(x)$$

be the perturbed polynomial. Assume that z_j is a simple root of $p(x)$. (Multiple roots can be considered in a similar manner. See, for example, [8].) Then

$$z_j(\epsilon) = z_j + \sum_{\ell=0}^{\infty} \gamma_{\ell} \epsilon^{\ell}$$

is a Taylor series for $z_j(\epsilon)$ as a function of ϵ , and $\gamma_1 = z'_j(0)$. Thus, $z_j(\epsilon) - z_j \approx z'_j(0)\epsilon$ is an approximation to the error in the roots due to perturbation of $p(x)$ by $\epsilon q(x)$. But $p(z_j(\epsilon)) + \epsilon q(z_j(\epsilon)) = 0$ and taking the derivative with respect to ϵ gives

$$p'(z_j(\epsilon))z'_j(\epsilon) + q(z_j(\epsilon)) + \epsilon q'(z_j(\epsilon))z'_j(\epsilon) = 0.$$

Therefore,

$$z'_j(\epsilon) = \frac{-q(z_j(\epsilon))}{p'(z_j(\epsilon)) + \epsilon q'(z_j(\epsilon))}.$$

Hence,

$$z_j(\epsilon) \approx z_j - \frac{q(z_j)}{p'(z_j)}\epsilon,$$

where $z_j(\epsilon)$ is the root of $p(x) + \epsilon q(x)$ and z_j is the root of $p(x)$. □

Example 2.19

(Conditioning of a Wilkinson polynomial of degree 7) Let

$$p(x) = \prod_{i=1}^7 (x - i) = x^7 - 28x^6 + 322x^5 + \cdots - 5040,$$

let $q(x) = x^6$, and let $\epsilon = -0.002$. (That is, we are introducing a small change in coefficient of x^6) of $p(x)$. Then

$$z_j(\epsilon) \approx z_j - \frac{q(z_j)\epsilon}{p'(z_j)} = j - \frac{j^6\epsilon}{\prod_{\ell \neq j} (j - \ell)} = j + \frac{j^6(0.002)}{\prod_{\ell \neq j}^7 (j - \ell)}.$$

Hence,

$$\begin{aligned} z_7(\epsilon) &\approx 7.33 \text{ for } z_7 = 7, \\ z_6(\epsilon) &\approx 5.22 \text{ for } z_6 = 6, \\ z_5(\epsilon) &\approx 5.65 \text{ for } z_5 = 5. \end{aligned}$$

Thus, the roots of p are unstable to small changes in the coefficients. The polynomial p is said to be ill-conditioned. We may need to employ double- or multiple-precision arithmetic to obtain better results, or even use interval arithmetic to check the results. $\square$

REMARK 2.25 Another method of finding roots of polynomials is to numerically find the eigenvalues of the *companion matrix*

$$C = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-1} \end{pmatrix}.$$

Roots of the polynomial $p(x) = x^n + a_{n-1}x^{n-1} + \dots + a_0$ are the same as the eigenvalues of C , since $\det(\lambda I - C) = p(\lambda)$. (See Chapter 5, starting on page 291, for a review of eigenvalues and eigenvectors, and a description of how to compute them numerically.) $\square$

REMARK 2.26 An alternate method, for finding rigorous bounds on all roots of a polynomial makes use of interval Newton methods combined with a systematic subdivision of a region in which all roots must lie. These techniques, special cases of *branch and bound* methods, are considered in Section 9.6.3 on page 523. $\square$

2.9 Additional Notes and Summary of Chapter 2

REMARK 2.27 Suppose that $f(x) = (x - z)^r q(x)$, where $r \geq 1$ and $\lim_{x \rightarrow z} q(x) \neq 0$. Function f is said to have the zero z of multiplicity r . Notice that if $r > 1$, then $f(z) = f'(z) = 0$. Hence, there may be convergence problems as $x_k \rightarrow z$ for Newton's method (and also the secant method), since

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

(Recall Theorem 2.5 on page 50, where we analyze the convergence of Newton's method. There, we required $|f'(x)| \geq m > 0$ for x near z . Also, see Exercise 22 on page 80.) Can we reduce this problem? Let $g(x) = f(x)/f'(x)$. Then

$$g(x) = \frac{(x-z)^r q(x)}{r(x-z)^{r-1}q(x) + (x-z)^r q'(x)} = \frac{(x-z)q(x)}{rq(x) + (x-z)q'(x)}.$$

Hence, $g(x)$ has a zero of multiplicity 1 at z and $g'(z) \neq 0$. If we apply Newton's method to $g(x)$ rather than to $f(x)$, our method is called the modified Newton's method, and works well for zeros of multiplicity greater than one. Thus,

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)} = x_k - \frac{f(x_k)f'(x_k)}{(f'(x_k))^2 - f(x_k)f''(x_k)}$$

is the modified Newton's method. □

Example 2.20

$$f(x) = x^4 - 4x^2 + 4 = (x - \sqrt{2})^2(x + \sqrt{2})^2.$$

k	x_k (Newton)	x_k (Modified Newton)
0	1.5	1.5
1	1.4583	1.4118
2	1.4366	1.4142
3	1.4255	.
4	1.4199	.
5	1.4171	.
$\vdots$	$\vdots$	
11	1.4143	
12	1.4142	

□

REMARK 2.28 Some of the advantages and disadvantages of the root-finding methods considered in this chapter appear in Table 2.2 (on page 84). □

2.10 Exercises

1. Consider the method of bisection applied to $f(x) = \arctan(x)$, with initial interval $x = [-4.9, 5.1]$.

- (a) Are the hypotheses under which the method of bisection converges valid? If so, then how many iterations would it take to obtain the solution to within an absolute error of 10^{-2} ?
 - (b) Apply Algorithm 2.1 with pencil and paper, until $k = 5$, arranging your computations carefully so you gain some intuition into the process.
2. Let f and $\mathbf{x}$ be as in Problem 1.
 - (a) Program Algorithm 2.1 in your chosen programming language. Print a_k , b_k , $f(a_k)$, $f(b_k)$, and $f(x_k)$ for each step, so you can see what is happening.
 - (b) Try to solve $f(x) = 0$ with $\epsilon = 10^{-2}$, $\epsilon = 10^{-4}$, $\epsilon = 10^{-8}$, $\epsilon = 10^{-16}$, $\epsilon = 10^{-32}$, $\epsilon = 10^{-64}$, and $\epsilon = 10^{-128}$.
 - i. For each ϵ , compute the k at which the algorithm should stop.
 - ii. What behavior do you actually observe in the algorithm? Can you explain this behavior?
3. Repeat Problem 2, but with $f(x) = x^2 - 2$ and initial interval $\mathbf{x} = [1, 2]$.
4. Use the program for the bisection method in Problem 2 to find an approximation to $1000^{\frac{1}{4}}$ which is correct to within 10^{-5} .
5. Suppose that $f \in C[a, b]$ has a unique zero $x^* \in [a, b]$, $f(a) < 0$ and $f(b) > 0$. Define the two sequences $\{x_k\}_{k=0}^{\infty}$ and $\{y_k\}_{k=0}^{\infty}$ by $x_0 = a$ and $y_0 = b$ and for $k = 1, 2, 3, \dots$ as follows.
 - (i) If $f\left(\frac{x_{k-1} + y_{k-1}}{2}\right) < 0$ then $x_k = \frac{x_{k-1} + y_{k-1}}{2}$ and $y_k = y_{k-1}$;
 - (ii) if $f\left(\frac{x_{k-1} + y_{k-1}}{2}\right) \geq 0$ then $x_k = x_{k-1}$ and $y_k = \frac{x_{k-1} + y_{k-1}}{2}$.

Prove that

 - (a) $x^* \in [x_k, y_k]$, $f(x_k) < 0$ and $f(y_k) \geq 0$ for $k = 0, 1, 2, \dots$
 - (b) $|x_k - x^*| \leq \frac{b-a}{2^k}$ for $k = 0, 1, 2, \dots$
6. Consider $g(x) = x - \arctan(x)$.
 - (a) Perform 10 iterations of the fixed point method $x_{k+1} = g(x_k)$, starting with $x = 5$, $x = -5$, $x = 1$, $x = -1$, and $x = 0.1$.
 - (b) Explain the behavior you observe in terms of the theory in this section.
7. $g(x) = \frac{x}{2} + \frac{a}{2x}$ has $x = \sqrt{a}$ as a fixed point. Suppose we use fixed point iteration, with a starting point $x_0 > \sqrt{a}$.

- (a) Use the theory from this section to determine a starting point x_0 , such that

$$|x_{k+1} - \sqrt{a}| \leq \frac{1}{2}|x_k - \sqrt{a}|,$$

for $k \geq 1$.

- (b) Perform a few iterations of the fixed point method on this problem (either by hand or with a short computer program).
- What is the convergence rate you observe? (That is, is it linear or quadratic? If it is linear then what is the constant by which the error is approximately multiplied on each iteration?)
 - Explain the behavior you observe in terms of the theory of this section.

8. To find a root of $f(x) = 0$ by iteration, rewrite the equation as

$$x = x + cf(x) = g(x)$$

for some constant $c \neq 0$. If α is a root of $f(x)$, if $f'(x)$ is continuous near $x = \alpha$, and if $f'(\alpha) \neq 0$, how should c be chosen to ensure that the sequence $x_{n+1} = g(x_n)$ converges to α , for x sufficiently close to α ?

9. Let α be a root of $f(x)$ and define an iteration formula by

$$x_{k+1} = z_{k+1} - \frac{f(z_{k+1})}{f'(x_k)}, \quad z_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

Show that the order of convergence of $\{x_k\}$ to α is at least 3.

10. It is desired to find the positive real root of the equation $x^3 + x^2 - 1 = 0$.

- Find an interval $x = [\underline{x}, \bar{x}]$ and a suitable fixed point iteration function $g(x)$ to accomplish this. Verify all conditions of the contraction mapping theorem.
- Find the minimum number of iterations n needed so that the absolute error in the n -th approximation to the root is correct to 10^{-4} . Also, use the fixed-point iteration method (with the g you determined in part (a)) to determine this positive real root accurate to within 10^{-4} .

11. Find an approximation to $1000^{\frac{1}{4}}$ correct to within 10^{-5} using the fixed point iteration method.

12. Consider the sequence defined by $x_{k+1} = b + \epsilon h^2(x_k)$ for $k = 0, 1, 2, \dots$, where $x_0 \in \mathbb{R}$ is given and $b, \epsilon \in \mathbb{R}$. Assume that

- $\exists M > 0$ such that, $|h(x)| \leq M$ for all $x \in \mathbb{R}$, and
- $\exists L > 0$ such that, $|h(v) - h(w)| \leq L|v - w|$ for all $v, w \in \mathbb{R}$.

Prove that if $2ML|\epsilon| < 1$, there is a unique $z \in \mathbb{R}$ such that $z = b + \epsilon h^2(z)$ and the sequence $\{x_k\}_{k=0}^\infty$ converges to this value z .

13. Consider the sequence $\{x_k\}_{k=0}^\infty$ defined by

$$x_{k+1} = g(x_k) = \gamma f(x_k) + h(x_k)$$

for $k = 0, 1, 2, \dots$, where $\gamma > 0$, $x_0 \in \mathbb{R}$, $f, h : \mathbb{R} \rightarrow \mathbb{R}$, and $|h'(x)| \leq \frac{3}{4}$ for all $x \in \mathbb{R}$. Assume that f is Lipschitz with Lipschitz constant L . Find a condition on γ that will guarantee convergence of the sequence $\{x_k\}_{k=0}^\infty$ to the unique value $z \in \mathbb{R}$ such that $z = g(z)$.

14. Let

$$g(x) = \frac{1}{3} \left[\frac{x^3}{3} - x^2 - \frac{5}{4}x + 4 \right]$$

and $G = [0, 2]$. Use the contraction mapping theorem to prove that if $x_0 \in G$, then the sequence defined by $x_{k+1} = g(x_k)$, $k = 0, 1, 2, \dots$, converges to the unique fixed point $z \in G$.

15. Consider the function $g(x) = e^{-x}$.

(a) Show that $g(x)$ has an unique fixed point $z \in (-\infty, \infty)$.

(b) Prove that g is a contraction on $[\ln 1.1, \ln 3]$.

(c) Prove that $g : [\ln 1.1, \ln 3] \rightarrow [\ln 1.1, \ln 3]$.

(d) Prove that $x_{k+1} = g(x_k)$ converges to the unique fixed point $z \in (-\infty, \infty)$ for any $x_0 \in (-\infty, \infty)$.

16. Assume that $f \in C^2(-\infty, \infty)$, $f(z) = 0$, and $S''(z) \neq 0$, where S is as in Remark 2.20 on page 68. Show that Steffensen's method (Formula 2.28 on page 68) is of second order. That is, show that Steffensen's method exhibits quadratic convergence.

17. Consider the fixed point iteration method $x_{k+1} = g(x_k)$, $k = 0, 1, \dots$ for solving the nonlinear equation $f(x) = 0$. Consider choosing an iteration function of the form

$$g(x) = x - af(x) - b(f(x))^2 - c(f(x))^3,$$

where a , b , and c are parameters to be determined. Find expressions for the parameters a , b , and c such that the iteration method is of fourth order.

18. Consider the iterative procedure

$$y_{j+1}^{(k+1)} = y_j + \frac{h}{2} \left[f(y_j) + f(y_{j+1}^{(k)}) \right], \quad \text{for } k = 0, 1, 2, \dots,$$

where $y_j \in \mathbb{R}$ is given, $f : \mathbb{R} \rightarrow \mathbb{R}$, and $y_{j+1}^{(0)} = y_j$. Suppose that f is Lipschitz on $\mathbb{R}$ with Lipschitz constant L . Prove that if $\frac{hL}{2} < 1$, then $y_{j+1}^{(k)} \rightarrow y_{j+1}$ as $k \rightarrow \infty$, where y_{j+1} satisfies

$$y_{j+1} = y_j + \frac{h}{2} [f(y_j) + f(y_{j+1})].$$

19. Assume that the equation $x^2 + bx + c = 0$ has two real roots α and β with $|\alpha| < |\beta|$. Show that the iteration method

$$x_{k+1} = -\frac{c}{x_k + b}$$

is convergent to α if x_0 is sufficiently close to α .

20. Consider $f(x) = \arctan(x)$. This function has a unique zero $z = 0$.

- (a) Find a radius ρ as in Theorem 2.5 (on page 50) such that the conclusions of Theorem 2.5 are true with $K_\rho(0) = [-\rho, \rho]$.
- (b) Use a digital computer with double precision arithmetic to do iterations of Newton's method, starting with $x_0 = 0.5, 1.0, 1.3, 1.4, 1.35, 1.375, 1.3875, 1.39375, 1.390625, 1.3921875$. Iterate until one of the following occurs:

- $|f(x)| \leq 10^{-10}$,
- an operation exception occurs, or
- 20 iterations are completed.

- (i) Describe the behavior you observe.
- (ii) Explain the behavior you observe in terms of the graph of f .
- (iii) Evidently, there is a point p such that, if $x_0 > p$, then Newton's method diverges, and if $x_0 < p$, then Newton's method converges.

(α) What would happen if $x_0 = p$ exactly? Illustrate what would happen on a graph of f .

(β) Do you think we could choose $x_0 = p$ exactly in practice?

21. Let $f(x) = x^2 - a$.

- (a) Write down and simplify the Newton's method iteration equation for $f(x) = 0$.
- (b) Assume $a > 1$. For $x_0 = a$, write down the constants M and m as in Theorem 2.5.
- (c) For $a = 2$, form a table of 15 iterations of Newton's method, starting with $x_0 = 2, x_0 = 4, x_0 = 8, x_0 = 16, x_0 = 32$, and $x_0 = 64$.
- (d) Explain your results in terms of the shape of the graph of f and in terms of the convergence theory in this section.

- (e) Compare your analysis here to the analysis you did for Problem 7 in this set and Example 2.9 on page 47.
22. Suppose z is a double root for f , i.e., $f(z) = f'(z) = 0 \neq f''(z)$. Show that if f'' is continuous and if Newton's method converges then it converges linearly in this case. In particular, $e_{n+1} \approx 0.5e_n$.
23. Let $f : [a, b] \rightarrow \mathbb{R}$ be a C^1 function satisfying $f'(x) \neq 0$ for $x \in [a, b]$. Let $\{x_k\}_{n=0}^\infty$ be a Newton iteration sequence for solving $f(x) = 0$, i.e., x_k satisfies the following equation:

$$x_k = x_{k-1} - \frac{f(x_{k-1})}{f'(x_{k-1})}.$$

Assume $x_k \in (a, b)$ and $\lim_{k \rightarrow \infty} x_k = r$. Show that $f(r) = 0$ and that, if $f'(x) \neq 0$ on $[a, b]$, then

$$|x_k - r| \leq \max_{x \in [a, b]} \frac{|f(x_k)|}{|f'(x)|}.$$

24. By writing $g'(x) = g'(z) + g''(c_x)(x - z)$, use the line of reasoning in Remark 2.11 (page 53) to show that Newton's method is quadratically convergent.
25. Using Newton's method, establish the iterative scheme

$$x_{k+1} = x_k(2 - Rx_k)$$

to calculate the reciprocal of a number R . Also, show that this iterative scheme yields quadratic convergence.

26. Describe Newton's method to compute the value of x that satisfies the equation $\int_0^x e^{t^2} dt = 1$.

27. Consider Newton's method $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$. Suppose that:

- (i) $f'(x)$ is continuous and $f'(x) > 0$ for $x \in \mathbb{R}$.
- (ii) $f(y) - f(x) \geq f'(x)(y - x)$ for $x, y \in \mathbb{R}$.
- (iii) $f(0) < 0$, $x_0 > 0$ and $f(x_0) > 0$.

Then,

- (a) Prove, by induction, that $x_0 \geq x_1 \geq x_2 \geq \cdots \geq x_{k+1} \geq \cdots \geq 0$.
- (b) Prove that $x_k \rightarrow z$ as $n \rightarrow \infty$, where z satisfies $f(z) = 0$.

28. *Hint: Use of Mathematica's interval arithmetic or the free INTLAB toolbox for MATLAB is recommended. There are also freely available Fortran 90 and C++ packages for interval arithmetic.*

- (a) Let f be as in Problem 20 of this set. Experiment with the interval Newton method for this problem, and with various intervals that contain zero. Explain what you have found.
 - (b) Use the interval Newton method to prove that there exists a unique solution to $f(x) = 0$ for $x \in [-1, 0]$, where $f(x) = x + e^x$.
 - (c) Iterate the interval Newton method to find as narrow bounds as possible on the solution proven to exist in part 28b.
 - (d) For comparison, attempt to use Theorem 2.5 (on page 50) to prove existence and uniqueness of a root of f within $[-1, 0]$.
29. An alternate form of interval Newton method is the *Krawczyk method*. The Krawczyk method is based on finding a fixed point of g , where

$$g(x) = x - f(x)/f'(\tilde{x}), \text{ where } \tilde{x} \text{ is some point in } \mathbf{x}.$$

The Krawczyk method is derived by using the mean value extension (See Problem 26 in Chapter 1.):

$$g(x) \in g(\tilde{x}) + \mathbf{g}'(\mathbf{x})(x - \tilde{x}), \text{ provided } x \in \mathbf{x} \text{ and } \tilde{x} \in \mathbf{x}.$$

Suppose:

- (a) $\mathbf{g}(\mathbf{x}) = \tilde{x} - \frac{f(\tilde{x})}{f'(\tilde{x})} + \left(1 - \frac{1}{f'(\tilde{x})} \mathbf{f}'(\mathbf{x})\right) (\mathbf{x} - \tilde{x}) \subseteq \mathbf{x};$
- (b) $\text{mag} \left(1 - \frac{\mathbf{f}'(\mathbf{x})}{f'(\tilde{x})}\right) = \max_{x \in \mathbf{x}} \left|1 - \frac{f'(x)}{f'(\tilde{x})}\right| < 1.$

Prove that there is a unique solution of $f(x) = 0$ in $\mathbf{x}$. (See page 156 for the formal definition of $\text{mag}(\mathbf{x})$.)

30. Show how Equation (2.19) follows from the derivation preceding it.
31. Repeat Exercise 20b, page 79, but with the secant method instead of Newton's method. (Use pairs of starting points $\{0.5, 1.0\}$, $\{1.0, 1.3\}$, etc.)
32. Verify that, in Equation (2.20) on page 63, $p(x_k) = f(x_k)$, $p(x_{k-1}) = f(x_{k-1})$, and $p(x_{k-2}) = f(x_{k-2})$.
33. Repeat Exercise 20, page 79, but with Müller's method instead of Newton's method. (Use triplets of starting points $\{0.5, 1.0, 1.3\}$, $\{1.0, 1.3, 1.4\}$, etc.)

34. Repeat Exercise 20b, page 79, but with Steffensen's method instead of Newton's method.
35. Do three steps of Newton's method, using complex arithmetic, for the function $f(z) = z^2 + 1$, with starting guess $z_0 = 0.2 + 0.7i$. Although you may use a computer program, you should show intermediate results, including z_k , $f(z_k)$, and $f'(z_k)$. (Note: Newton's method with complex arithmetic can be viewed as a multivariate Newton method in two variables; see Exercise 9 on page 483, in Section 8.3.)
36. Example 2.19 deals with the Wilkinson³ polynomial of degree 7, where the Wilkinson polynomial of degree n is defined to be

$$p_n(x) = \prod_{i=1}^n (x - i).$$

These polynomials have roots that are notoriously sensitive to the coefficients of the polynomial in power form, so they are interesting test problems.

- (a) Find a_0 through a_7 exactly, as integers (by using, say, the “**Expand**” function in Mathematica, or something similar in your favorite symbolic manipulation program, or else by looking them up.)
- (b) Program Algorithm 2.3 (on page 72). Here, you will need to make some choices, such as the method you will use in Step 1 and how you will choose the starting points.⁴
- (c) Test your program from part 36b on several simple examples, such as $p(x) = x^2 + 3x + 2$ or $p(x) = x^3 - 1$, to make sure it is working correctly.
- (d) Using the coefficients from part 36a, try your program on the Wilkinson polynomial. Do you get the same thing as Example 2.19?
- (e) Try your program for the Wilkinson polynomial of degree $n = 20$ (after repeating part 36a for $n = 20$.) Do you get reasonable answers?

³J. H. Wilkinson was a famous numerical analyst in the middle of the twentieth century, at the beginning of the era of modern digital computers. He invented the concept of “backward error analysis,” that is, the technique of showing that roundoff errors produce an answer that is the exact answer to a nearby problem. He is also a father of modern numerical linear algebra; one of his works is the monograph *The Algebraic Eigenvalue Problem* [101]. The Society of Industrial and Applied Mathematicians awards the “Wilkinson Prize,” in honor of J. H. Wilkinson, for outstanding numerical software packages.

⁴One way to choose the starting points is to compute R as in “Result 1” on page 70, and choose the initial guess in the disk.

- (f) Do a stability analysis similar to Example 2.19, for the $n = 20$ case. Is your analysis consistent with your results from part 36e? (Note that, although the coefficients are integers, some of them have more digits than can be represented internally in an IEEE double precision number, so that the polynomial that the computer is actually storing in its memory is a perturbation of the Wilkinson polynomial. Estimate the size of that perturbation.)
37. Write a program that combines Newton's, Horner's, and Müller's methods to obtain *all* the roots of the following polynomials.
- (I) $P(x) = x^4 - 1$.
- (II) $P(x) = x^5 - 5x^4 - 8x^3 + 40x^2 - 9x + 45$.
- (III) $P(x) = 1.1x^5 - 5x^4 - 8x^3 + 40x^2 - 9x + 45$.

Use the following steps in your implementation.

- Repeat the following until *all* the approximate roots of $P(x)$ are determined.
 - (a) Employ Müller's method to determine a root z of $P(x)$.
 - (b) Deflate $P(x) = (x - z)Q(x)$ using Horner's algorithm. (That is, use Algorithm 2.3 on page 72.)
 - (c) Replace $P(x)$ by $Q(x)$.
 - (d) Go to Step (a).
- Using each approximate root obtained as an initial guess, apply Newton's method to obtain better approximations for the roots of the original polynomial.

Explain your observation for the answers obtained for (II) and (III).

38. Does your program from Problem 37 work well for the Wilkinson polynomial (Example 2.19 on page 73) of degree 7? What about degree 10? What about degree 20?

TABLE 2.2: A summary of the methods considered in Chapter 2.

Method	Advantages	Disadvantages	Other Comments
Bisection	simple and reliable; converges provided that $f(a)f(b) < 0$ and f is continuous	not rapidly convergent	
Fixed-point	simple; is a useful theoretical tool for analyzing other methods	If $g'(z) \neq 0$ then linearly convergent. Also, may be difficult to find a good g . Generally, requires $g : G \rightarrow G$ and g a contraction.	is connected to other methods and theory throughout numerical analysis, analysis, topology, and differential equations
Newton's	generally quadratically convergent	$f'(x)$ is required; may be only convergent in a small interval about z	special fixed-point scheme with $g(x) = x - f(x)/f'(x)$. (Note that $g'(z) = 0$.)
Interval Newton	Often quadratically convergent; provides mathematically rigorous proof of existence and uniqueness, and provides mathematically rigorous bounds on the error	Requires interval arithmetic; convergence radius may be smaller than the point Newton method	When combined with intersection of the previous iterate, convergence may occur when the point Newton method diverges.
Secant	generally converges superlinearly; $f'(x)$ not required	may only have a small interval of convergence	not a fixed-point scheme
Müller's	often converges superlinearly; useful for finding complex roots	may only have a small interval of convergence	not a fixed-point scheme
Modified Newton's	quadratically convergent for multiple roots	$f'(x)$ and $f''(x)$ are required	special fixed-point scheme (Newton's method applied to $g(x) = f(x)/f'(x)$)
Aitken's	simple method that speeds up linearly convergent sequences	sequence must be linearly convergent	

Chapter 3

Numerical Linear Algebra

Numerical linear algebra is primarily concerned with two important subjects:

1. efficient solution of linear systems, and
2. computation of eigenvalues and eigenvectors of matrices.

Numerical solution of nonlinear systems, partial differential equations, integral equations, etc., generally involve the solution of linear systems. Eigenvalue-eigenvector computation occurs in physical and biological applications. Also, for example, solution of systems of differential equations can involve eigenvalue-eigenvector computation.

This chapter deals primarily with efficient solution of linear systems, while eigenvalues and eigenvectors are treated in Chapter 5. Some good reference texts for numerical linear algebra are [34, 37, 40, 68, 78, 85, 97, 103].

3.1 Basic Results from Linear Algebra

Here, we give a brief review of matrix algebra.¹

DEFINITION 3.1 Let $\mathbb{R}^n(\mathbb{C}^n)$ be the n -dimensional space of n -tuples of real (complex) numbers, i.e., if $x \in \mathbb{C}^n$ then $x = [x_1, x_2, \dots, x_n]^T$ where $x_i \in \mathbb{C}$ for $i = 1, 2, \dots, n$.

DEFINITION 3.2 $x^T = [x_1, x_2, \dots, x_n]$, $x^H = [\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n]$, where T and H refer to transpose and conjugate transpose. (If $x = a + ib$, then $\bar{x} = a - ib$.)

REMARK 3.1 $x^H x$ is a nonnegative number whereas $x x^H$ is an $n \times n$ matrix. □

¹This is not meant to be a self-contained introduction, but introduces our notation, and serves as a quick reference. It is assumed that the reader already knows the basic concepts.

DEFINITION 3.3 The set of real (complex) $n \times m$ matrices is denoted by $L(\mathbb{R}^m, \mathbb{R}^n)$ ($L(\mathbb{C}^m, \mathbb{C}^n)$) or $L(\mathbb{R}^n)$ ($L(\mathbb{C}^n)$) if $m = n$. The elements of matrix A will be written a_{ij} , and we sometimes write $A = (a_{ij})$. We also may sometimes say $A \in \mathbb{R}^{m \times n}$ to mean that A is a real m by n matrix, and $A \in \mathbb{C}^{m \times n}$ to mean that A is a complex m by n matrix.

DEFINITION 3.4 If $A = (a_{ij})$, then $A^T = (a_{ji})$ and $A^H = (\bar{a}_{ji})$ denote the transpose and conjugate transpose of A , respectively.

REMARK 3.2 $(AB)^H = B^H A^H$, $(AB)^T = B^T A^T$ and if A is $m \times n$ then A^T or A^H is $n \times m$. $\square$

DEFINITION 3.5 If A is an $m \times n$ matrix and B is an $n \times p$ matrix, then $C = AB$ where

$$c_{ij} = \sum_{k=1}^n a_{ik} b_{kj}$$

for $i = 1, \dots, m$, $j = 1, \dots, p$. Thus, C is an $m \times p$ matrix.

DEFINITION 3.6 If A is an $n \times n$ matrix ($A \in L(\mathbb{C}^n)$), then a scalar λ and a nonzero x are an eigenvalue and eigenvector of A if $Ax = \lambda x$.

DEFINITION 3.7 Suppose $A \in L(\mathbb{R}^n)$ or $A \in L(\mathbb{C}^n)$. A^{-1} is the inverse of A if $A^{-1}A = AA^{-1} = I$, where I is the n by n identity matrix, consisting of 1's on the diagonal and 0's in all off-diagonal elements. If A has an inverse, then A is said to be nonsingular or invertible.

DEFINITION 3.8 The rank of a matrix A , $\text{rank}(A)$, is the maximum number of linearly independent rows it possesses. It can be shown that this is the same as the maximum number of linearly independent columns. If A is an m by n matrix and $\text{rank}(A) = \min\{m, n\}$, then A is said to be of full rank. For example, if $m < n$ and the rows of A are linearly independent, then A is of full rank.

The following theorem deals with rank, nonsingularity, and solutions to systems of equations.

THEOREM 3.1

Let A be an $n \times n$ matrix ($A \in L(\mathbb{C}^n)$). Then the following are equivalent:

1. A is nonsingular.
2. $\det(A) \neq 0$, where $\det(A)$ is the determinant of the matrix A .

3. The linear system $Ax = 0$ has only the solution $x = 0$.
4. For any $b \in \mathbb{C}^n$, the linear system $Ax = b$ has a unique solution.
5. The columns (and rows) of A are linearly independent, that is, if $a_1, a_2, \dots, a_n$ are the columns of A and $\sum_{i=1}^n \beta_i a_i = 0$, then $\beta_i = 0$ for $i = 1, 2, \dots, n$. (That is, $\text{rank}(A) = n$.)

REMARK 3.3 By Definition 3.6 and Theorem 3.1, λ is an eigenvalue of A if and only if $\det(A - \lambda I) = 0$. The equation $\det(A - \lambda I) = 0$ is called the characteristic equation of A . $\square$

DEFINITION 3.9 $\rho(A) = \max_{1 \leq i \leq n} |\lambda_i|$, where $\{\lambda_i\}_{i=1}^n$ is the set of eigenvalues of A , is called the spectral radius of A .

DEFINITION 3.10 If $A^T = A$, then A is said to be symmetric. If $A^H = A$, then A is said to be Hermitian.

Example 3.1

If $A = \begin{pmatrix} 1 & 2-i \\ 2+i & 3 \end{pmatrix}$, then $A^H = \begin{pmatrix} 1 & 2-i \\ 2+i & 3 \end{pmatrix}$, so A is Hermitian. $\square$

DEFINITION 3.11 For $A \in L(\mathbb{R}^n)$, if $A^T = A$ and $x^T A x > 0$ for any $x \in \mathbb{R}^n$ except $x = 0$, then A is said to be symmetric positive definite. For $A \in L(\mathbb{C}^n)$, if $A^H = A$ and $x^H A x > 0$ for $x \in \mathbb{C}^n$, $x \neq 0$, then A is said to be Hermitian positive definite.² Similarly, if $x^T A x \geq 0$ (for a real matrix A) or $x^H A x \geq 0$ (for a complex matrix A) for every $x \neq 0$, we say that A is symmetric positive semi-definite or Hermitian positive semi-definite, respectively.

Example 3.2

If $A = \begin{pmatrix} 4 & 1 \\ 1 & 3 \end{pmatrix}$, then $A^T = A$, so A is symmetric.

²A matrix need not be symmetric or Hermitian to be positive definite or positive semi-definite, although that is the usual context. That is, A is positive definite provided $x^T A x > 0$ for every $x \neq 0$, and positive semi-definite provided $x^T A x \geq 0$ for every $x \neq 0$.

Also, $x^T Ax = 4x_1^2 + 2x_1x_2 + 3x_2^2 = 3x_1^2 + (x_1 + x_2)^2 + 2x_2^2 > 0$ for $x \neq 0$. Thus, A is symmetric positive definite. $\square$

PROPOSITION 3.1

If A is symmetric positive definite or Hermitian positive definite, then its eigenvalues are real and positive.

PROOF Suppose that $Ax = \lambda x$. Then $x^H Ax = \lambda x^H x$. Also, $(x^H Ax)^H = (\lambda x^H x)^H$, which yields $x^H Ax = \bar{\lambda} x^H x$. Thus, $\bar{\lambda} x^H x = \lambda x^H x$. Hence $\bar{\lambda} = \lambda$, so λ is real. Also, $\lambda = x^H Ax / x^H x > 0$. $\square$

Now consider a linear system of equations $Ax = b$, where A is $n \times n$, and $b, x \in \mathbb{R}^n$.

DEFINITION 3.12 *Elementary row operations on a system of linear equations are of the following three types:*

1. *interchanging two equations,*
2. *multiplying an equation by a nonzero number,*
3. *adding to one equation a scalar multiple of another equation.*

THEOREM 3.2

If system $Bx = d$ is obtained from system $Ax = b$ by a finite sequence of elementary operations, then the two systems have the same solutions.

A proof of Theorem 3.2 can be found in elementary texts on linear algebra and can be done, for example, with Theorem 3.1 and using elementary properties of determinants.

3.2 Normed Linear Spaces

We will use these concepts both in this chapter and in subsequent chapters.

DEFINITION 3.13 *Let V be a vector space (recall that a vector space is closed under addition and scalar multiplication) over the field of real or complex numbers. V is called a normed vector space if to each $u \in V$ a nonnegative number $\|u\|$, called the norm of u , is assigned with the following properties:*

1. $\|u\| \geq 0$.
2. $\|u\| = 0$ if and only if $u = 0$.
3. $\|\lambda u\| = |\lambda| \|u\|$ for $\lambda \in \mathbb{R}$ (or $\lambda \in \mathbb{C}$ if V is a complex vector space).
4. $\|u + v\| \leq \|u\| + \|v\|$ (triangle inequality).

DEFINITION 3.14 *W is a subspace of a real vector space V if $u \in W$, $v \in W$ imply that $\alpha u + \beta v \in W$ for all $\alpha, \beta \in \mathbb{R}$.*

DEFINITION 3.15 *Let V be a vector space. Then $u_1, u_2, \dots, u_n \in V$ are linearly independent if $\sum_{i=1}^n \alpha_i u_i = 0$ implies that $\alpha_1 = \alpha_2 = \dots = \alpha_n = 0$.*

Example 3.3

Let $V = C[a, b]$, the space of continuous functions on an interval $[a, b]$. Then $u_1 = 1$, $u_2 = x$ are linearly independent, while $u_1 = 1$, $u_2 = x$, $u_3 = 2 - x$ are linearly dependent. $\square$

DEFINITION 3.16 *Let $u_1, u_2, \dots, u_n \in V$. The set of all linear combinations of $u_1, u_2, \dots, u_n$ is called the span of $u_1, u_2, \dots, u_n$.*

REMARK 3.4 Let $W = \text{span}(u_1, u_2, \dots, u_n)$. It is easy to show that W is a subspace of V . ($w \in W$ has the form $w = \sum_{i=1}^n c_i u_i$.) $\square$

DEFINITION 3.17 *If $V = \text{span}(u_1, u_2, \dots, u_n)$ and $u_1, u_2, \dots, u_n$ are linearly independent, then $u_1, u_2, \dots, u_n$ forms a basis for V .*

REMARK 3.5 Suppose that V has a basis of n elements. Then, every basis of V has n elements. Any collection of $n + 1$ elements is linearly dependent. $\square$

DEFINITION 3.18 *If a vector space V has a basis with a finite number n of elements, then n is called the dimension of the vector space.*

Example 3.4

Let P^2 represent the set of polynomials of degree 2 or less. Then P^2 is a subspace of $V = C[a, b]$. $P^2 = \text{span}(1, x, x^2)$, and, since 1, x , and x^2 are linearly independent, the dimension of P^2 is 3. $\square$

Consider $V = \mathbb{C}^n$, the vector space of n -tuples of complex numbers. Note that $x \in \mathbb{C}^n$ has the form $x = (x_1, x_2, \dots, x_n)^T$. Also,

$$x + y = (x_1 + y_1, x_2 + y_2, \dots, x_n + y_n)^T$$

and

$$\lambda x = (\lambda x_1, \lambda x_2, \dots, \lambda x_n)^T.$$

Important norms on $\mathbb{C}^n$ are:

$$(a) \quad \|x\|_\infty = \max_{1 \leq i \leq n} |x_i|: \quad \text{the } \ell_\infty \text{ or max norm (for } z = a + ib, |z| = \sqrt{a^2 + b^2} = \sqrt{z\bar{z}}.)$$

$$(b) \quad \|x\|_1 = \sum_{i=1}^n |x_i|: \quad \text{the } \ell_1 \text{ norm}$$

$$(c) \quad \|x\|_2 = \left(\sum_{i=1}^n |x_i|^2 \right)^{1/2} : \quad \text{the } \ell_2 \text{ norm (Euclidean norm)}$$

$$(d) \quad \|x\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{1/p} : \quad \text{the } \ell_p \text{ norm}$$

REMARK 3.6 (b) and (c) are special cases of (d), and it can be shown that $\|x\|_\infty = \lim_{p \rightarrow \infty} \left(\sum_{i=1}^n |x_i|^p \right)^{1/p}$. □

REMARK 3.7 It is not hard to see that (1), (2), and (3) of Definition 3.13 are satisfied for the above norms. The triangle inequality (4) of Definition 3.13 for (a) and (b) follow from $|x_i + y_i| \leq |x_i| + |y_i|$, e.g.,

$$\|x + y\|_1 = \sum_{i=1}^n |x_i + y_i| \leq \sum_{i=1}^n |x_i| + \sum_{i=1}^n |y_i| = \|x\|_1 + \|y\|_1.$$

□

The triangle inequality, or Minkowski's inequality, is not proved here for p -norms since we won't use general p -norms. The triangle inequality for the Euclidean norm follows from:

THEOREM 3.3

(the Cauchy-Schwarz inequality)

$$\left| \sum_{i=1}^n x_i \bar{y}_i \right| \leq \|x\|_2 \|y\|_2.$$

PROOF Let $|x|$ and $|y|$ be vectors with components $|x_i|$ and $|y_i|$. Then for $\theta \in \mathbb{R}$,

$$0 \leq \|\theta|x| + |y|\|_2^2 = \theta^2 \sum_{j=1}^n |x_j|^2 + 2\theta \sum_{j=1}^n |x_j||y_j| + \sum_{j=1}^n |y_j|^2.$$

The quadratic polynomial in θ on the right-hand side does not change sign. Note that if $p(\theta) = a\theta^2 + b\theta + c \geq 0$ then the discriminant $b^2 - 4ac \leq 0$. Thus,

$$\left(\sum_{j=1}^n |x_j||y_j| \right)^2 \leq \sum_{j=1}^n |x_j|^2 \sum_{j=1}^n |y_j|^2.$$

But

$$\left| \sum_{j=1}^n x_j \bar{y}_j \right|^2 \leq \left(\sum_{j=1}^n |x_j||y_j| \right)^2 \leq \sum_{j=1}^n |x_j|^2 \sum_{j=1}^n |y_j|^2,$$

that is,

$$\left| \sum_{j=1}^n x_j \bar{y}_j \right| \leq \|x\|_2 \|y\|_2.$$

□

The triangle inequality for $\|\cdot\|_2$ now follows from the Cauchy–Schwarz inequality as follows:

$$\begin{aligned} \|x + y\|_2 &= \left(\sum_{j=1}^n (x_j + y_j)(\bar{x}_j + \bar{y}_j) \right)^{1/2} \\ &= \left(\sum_{j=1}^n |x_j|^2 + \sum_{j=1}^n (x_j \bar{y}_j + \bar{x}_j y_j) + \sum_{j=1}^n |y_j|^2 \right)^{1/2} \\ &\leq \left(\|x\|_2^2 + 2 \left| \sum_{j=1}^n x_j \bar{y}_j \right| + \|y\|_2^2 \right)^{1/2} \\ &\leq (\|x\|_2^2 + 2\|x\|_2 \|y\|_2 + \|y\|_2^2)^{1/2} = \|x\|_2 + \|y\|_2. \end{aligned}$$

An important type of normed space is an inner product space:

DEFINITION 3.19 An inner product on $\mathbb{C}^n \times \mathbb{C}^n$ is a complex-valued function, denoted by $(\cdot, \cdot)$ defined on all pairs $x, y \in \mathbb{C}^n$ such that

- (a) $(x, x) \geq 0$ and $(x, x) = 0$ if and only if $x = 0$,

(b) $(x, y) = \overline{(y, x)},$

(c) $(x, \lambda y + \mu z) = \overline{\lambda}(x, y) + \overline{\mu}(x, z)$ for $\lambda, \mu \in \mathbb{C}.$

REMARK 3.8 $(\lambda x, y) = \overline{(y, \lambda x)} = \lambda(x, y).$ □

REMARK 3.9 If $\|x\| = (x, x)^{1/2}$, then $\|\cdot\|$ is a norm. Thus, any inner product space is also a normed vector space. □

Example 3.5

The following define inner products.

1. $(x, y) = \langle x, y \rangle = \sum_{i=1}^n x_i \overline{y}_i.$ Clearly, $\langle x, x \rangle = \|x\|_2^2.$ Also, $\langle x, y \rangle = y^H x$ where $y^H = (\overline{y}_1, \overline{y}_2, \dots, \overline{y}_n).$

2. $(x, y) = \sum_{j=1}^n \sum_{i=1}^n x_j h_{ji} \overline{y}_i = \langle Hx, y \rangle = y^H Hx$ where H is an $n \times n$ Hermitian positive definite matrix. □

We now introduce a concept and notation for describing errors in computations involving vectors.

DEFINITION 3.20 The distance from u to v is defined as $\|u - v\|.$

The following concept is also fundamental to inner product spaces. In particular, we will use it heavily in Section 3.3.8 and Section 4.2.2.

DEFINITION 3.21 Let $(\cdot, \cdot)$ represent an inner product. Two vectors u and v are called orthogonal with respect to $(\cdot, \cdot)$ provided $(u, v) = 0.$ A set of vectors $v^{(i)}$ is said to be orthonormal, provided $(v^{(i)}, v^{(j)}) = \delta_{ij},$ where δ_{ij} is the Kronecker delta function

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

To analyze iterative techniques involving vectors and matrices, we use:

DEFINITION 3.22 A sequence of vectors $\{x^k\}_{k=1}^\infty$ is said to converge to a vector $x \in \mathbb{C}^n$ if and only if $\|x^k - x\| \rightarrow 0$ as $k \rightarrow \infty$ for some norm $\|\cdot\|.$

REMARK 3.10 Definition 3.22 implies that a sequence of vectors $\{x^k\}$ converges to x if and only if³ $x_i^k \rightarrow x_i$ as $k \rightarrow \infty$ for all i . (We will discuss this further later.) $\square$

Example 3.6

Let $x^k = [1^k, (\frac{1}{2})^k, (\frac{1}{3})^k, (\frac{1}{4})^k]^T$. Then $x^k \rightarrow x = [1, 0, 0, 0]^T$ in the ℓ_∞ - norm, since

$$\|x^k - x\|_\infty = \max_{1 \leq i \leq n} |x_i^k - x_i| = \left(\frac{1}{2}\right)^k \rightarrow 0 \text{ as } k \rightarrow \infty.$$

$\square$

DEFINITION 3.23 Two norms $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$ are called equivalent if there exist positive constants c_1 and c_2 and such that

$$c_1 \|x\|_\alpha \leq \|x\|_\beta \leq c_2 \|x\|_\alpha.$$

Hence, also,

$$\frac{1}{c_2} \|x\|_\beta \leq \|x\|_\alpha \leq \frac{1}{c_1} \|x\|_\beta.$$

REMARK 3.11 If the norms $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$ are equivalent, then $x^k \rightarrow x$ in norm $\|\cdot\|_\alpha$ if and only if $x^k \rightarrow x$ in norm $\|\cdot\|_\beta$. $\square$

THEOREM 3.4

Any two norms on $\mathbb{C}^n$ are equivalent.

REMARK 3.12 By Theorem 3.4, convergence in any norm in $\mathbb{C}^n$ thus implies convergence in any other norm. $\square$

PROOF (of Theorem 3.4) We will prove that any norm is equivalent to the ℓ_2 norm. Then any two norms are equivalent. That is, if

$$c_1 \|x\|_2 \leq \|x\|_\beta \leq c_2 \|x\|_2$$

and

$$c_1^* \|x\|_2 \leq \|x\|_\alpha \leq c_2^* \|x\|_2,$$

then

$$\frac{c_1}{c_2^*} \|x\|_\alpha \leq \|x\|_\beta \leq \frac{c_2}{c_1^*} \|x\|_\alpha.$$

³Here, we are considering only finite-dimensional vector spaces.

Let $\|\cdot\|$ be any vector norm. Let $e_1, e_2, \dots, e_n$ be the usual basis vectors for $\mathbb{C}^n$, that is, $e_j = (\delta_{1j}, \delta_{2j}, \dots, \delta_{nj})^T$, $j = 1, 2, \dots, n$, where

$$\delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$$

Then

$$x = (x_1, x_2, \dots, x_n)^T = \sum_{j=1}^n x_j e_j.$$

Thus,

$$\|x\| \leq \sum_{j=1}^n |x_j| \|e_j\| \leq \left(\sum_{j=1}^n |x_j|^2 \right)^{\frac{1}{2}} \left(\sum_{j=1}^n \|e_j\|^2 \right)^{\frac{1}{2}}$$

by the Cauchy-Schwarz inequality. Thus,

$$\|x\| \leq \gamma_1 \|x\|_2, \text{ where } \gamma_1 = \left(\sum_{j=1}^n \|e_j\|^2 \right)^{\frac{1}{2}}.$$

Let $h(x) = \|x\|$, i.e., $h: \mathbb{C}^n \rightarrow \mathbb{R}$. Notice that

$$|h(x) - h(y)| = |\|x\| - \|y\|| \leq \|x - y\| \leq \gamma_1 \|x - y\|_2.$$

Let $S = \{x \in \mathbb{C}^n : \|x\|_2 = 1\}$, i.e., S is the surface of the unit ball in $\mathbb{C}^n$. The unit spherical surface S is closed and bounded, and h is a continuous function on S . By a classical theorem of topology or analysis,⁴ h has a minimum at some z in S . Let $\gamma_0 = h(z) \leq h(y)$ for all $y \in S$. (Notice that $\gamma_0 > 0$ since if $\gamma_0 = 0$ then $h(z) = \|z\| = 0$ implies $z = 0$, which is impossible since $\|z\|_2 = 1$.) Now $y = x/\|x\|_2 \in S$. Thus, $0 < \gamma_0 \leq h(x/\|x\|_2) = \|x\|/\|x\|_2$. Hence, $\gamma_0 \|x\|_2 \leq \|x\|$. Combining this with our earlier result $\|x\| \leq \gamma_1 \|x\|_2$ gives $\gamma_0 \|x\|_2 \leq \|x\| \leq \gamma_1 \|x\|_2$. $\square$

REMARK 3.13 Recall that earlier we claimed that $\|x - x^k\| \rightarrow 0$ as $k \rightarrow \infty$ if and only if $x_i^k \rightarrow x_i$, $1 \leq i \leq n$. This is now obvious, since there exist constants c_1 and c_2 such that $c_1 \|x - x^k\|_\infty \leq \|x - x^k\| \leq c_2 \|x - x^k\|_\infty$, keeping in mind that $\|x - x^k\|_\infty = \max_{1 \leq i \leq n} |x_i - x_i^k|$. $\square$

Consider a specific case of Theorem 3.4:

PROPOSITION 3.2

For each $x \in \mathbb{C}^n$, $\|x\|_\infty \leq \|x\|_2 \leq \sqrt{n} \|x\|_\infty$.

⁴The theorem states that a continuous function on a compact set attains its minimum and a maximum at points on that set.

PROOF Let x_j be the element of x such that

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i| = |x_j|.$$

Then

$$\|x\|_\infty^2 = |x_j|^2 \leq \sum_{i=1}^n |x_i|^2 = \|x\|_2^2.$$

Also,

$$\|x\|_2^2 = \sum_{i=1}^n |x_i|^2 \leq \sum_{i=1}^n |x_j|^2 = n|x_j|^2 = n\|x\|_\infty^2.$$

Thus, $\|x\|_\infty \leq \|x\|_2 \leq \sqrt{n} \|x\|_\infty$. □

We now consider norms for $n \times n$ complex matrices. In the following, A and B are arbitrary square matrices and λ is a complex number.

DEFINITION 3.24 A matrix norm is a real-valued function of A , denoted by $\|\cdot\|$ satisfying:

1. $\|A\| \geq 0$.
2. $\|A\| = 0$ if and only if $A = 0$.
3. $\|\lambda A\| = |\lambda| \|A\|$.
4. $\|A + B\| \leq \|A\| + \|B\|$.
5. $\|AB\| \leq \|A\| \|B\|$.

REMARK 3.14 In contrast to vector norms, we have an additional fifth property, referred to as a submultiplicative property, dealing with the norm of the product of two matrices. □

The following will be used in our analysis of iterative methods for solving systems of equations, and elsewhere.

PROPOSITION 3.3

If $\|\cdot\|$ is any matrix norm as in Definition 3.24,

$$\rho(A) \leq \|A\|.$$

PROOF Let

$$B = (x, x, x, \dots, x),$$

that is, let B be the n by n matrix, each of whose columns is the nonzero vector x , where x is such that $Ax = \lambda x$. Then

$$AB = (\lambda x, \dots \lambda x) = \lambda B.$$

Thus, by the third and fifth properties of matrix norms, $|\lambda| \|B\| \leq \|A\| \|B\|$, so $|\lambda| \leq \|A\|$ for any eigenvalue of A . Since the spectral radius is the maximum eigenvalue of A , the result follows. $\square$

REMARK 3.15 By choosing a particular ordering of the elements, $n \times n$ matrices may be viewed as vectors in $\mathbb{C}^{n^2}$, e.g., $\tilde{a}_\ell = a_{jk}$, $\ell = j + n(k-1)$ for $j = 1, 2, \dots, n$ and $k = 1, 2, \dots, n$. Thus, if we define a matrix norm using a vector norm on the n^2 vector, the first four conditions in the above definition are automatically satisfied. However, in general, condition (5) will not hold. For example, consider $\|A\|_\infty = \max_{i,j} |a_{ij}|$, and take

$$A = B = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

Then $\|AB\|_\infty \not\leq \|A\|_\infty \|B\|_\infty$. One norm for which this procedure holds is the Euclidean norm

$$\|A\|_E = \left(\sum_{i,j=1}^n |a_{ij}|^2 \right)^{\frac{1}{2}},$$

which is also called the *Frobenius norm*. One sees that (1), (2), (3), and (4) of Definition 3.24 are satisfied by considering the corresponding ℓ_2 -norm $\|\cdot\|_2$ of the corresponding vector in $\mathbb{C}^{n^2}$. To prove (5), we have $\square$

$$\|AB\|_E^2 = \sum_{i,j=1}^n \left(\sum_{k=1}^n a_{ik} b_{kj} \right)^2 \leq \sum_{i,j=1}^n \sum_{k=1}^n |a_{ik}|^2 \sum_{k=1}^n |b_{kj}|^2 = \|A\|_E^2 \|B\|_E^2.$$

DEFINITION 3.25 A matrix norm $\|A\|$ and a vector norm $\|x\|$ are called compatible if for all vectors x and matrices A we have $\|Ax\| \leq \|A\| \|x\|$.

REMARK 3.16 A consequence of the Cauchy-Schwarz inequality is that $\|Ax\|_2 \leq \|A\|_E \|x\|_2$, i.e., the Euclidean norm $\|\cdot\|_E$ for matrices is compatible with the ℓ_2 -norm $\|\cdot\|_2$ for vectors. $\square$

We now define some commonly used matrix norms.

DEFINITION 3.26 Given a vector norm $\|\cdot\|$, we define a natural or induced matrix norm associated with it as

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}. \quad (3.1)$$

REMARK 3.17 If an induced matrix norm is compatible with the given vector norm, then

$$\|A\| \|x\| \geq \|Ax\| \text{ for all } x \in \mathbb{C}^n. \quad (3.2)$$

□

REMARK 3.18 It is straightforward to show that an induced matrix norm satisfies the five properties required of a matrix norm so is indeed a matrix norm. In particular,

$$\|AB\| = \sup_{x \neq 0} \frac{\|ABx\|}{\|x\|} \leq \sup_{x \neq 0} \frac{\|A\| \|Bx\|}{\|x\|} = \|A\| \|B\|,$$

since $\|ABx\| = \|Az\| \leq \|A\| \|z\|$.

□

REMARK 3.19 Definition 3.26 is equivalent to

$$\|A\| = \sup_{\|y\|=1} \|Ay\|,$$

since

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \sup_{x \neq 0} \left\| A \frac{x}{\|x\|} \right\| = \sup_{\|y\|=1} \|Ay\|$$

(letting $y = x/\|x\|$).

□

REMARK 3.20 We shall use matrix norms $\|\cdot\|_\infty$, $\|\cdot\|_1$, and $\|\cdot\|_2$ induced by the corresponding vector norms. That is,

$$\|A\|_\infty = \sup_{x \neq 0} \frac{\|Ax\|_\infty}{\|x\|_\infty}, \quad \|A\|_1 = \sup_{x \neq 0} \frac{\|Ax\|_1}{\|x\|_1}, \quad \text{and} \quad \|A\|_2 = \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}.$$

□

REMARK 3.21 The Frobenius norm $\|\cdot\|_F$ is not a natural norm, because $\|I\| = 1$ for any natural norm but $\|I\|_F = \sqrt{n}$. Consider briefly the relation between $\|A\|_2$ and $\|A\|_F$. First note that, in general, $\|A\|_2 \neq \|A\|_F$. (For example, let $A = I$.) Also,

$$\|A\|_2 = \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} \leq \sup_{x \neq 0} \frac{\|A\|_F \|x\|_2}{\|x\|_2} = \|A\|_F.$$

Thus, $\|A\|_2 \leq \|A\|_E$. □

We now develop explicit expressions for $\|A\|_\infty$, $\|A\|_1$, and $\|A\|_2$.

PROPOSITION 3.4

(Formulas for the induced ℓ_1 and ℓ_∞ matrix norms)

$$(a) \|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}| = \{\text{maximum absolute row sum}\}.$$

$$(b) \|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^n |a_{ij}| = \{\text{maximum absolute column sum}\}.$$

PROOF

$$\|Ax\|_\infty = \max_{1 \leq i \leq n} \left| \sum_{j=1}^n a_{ij} x_j \right| \leq \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}| |x_j| \leq \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}| \|x\|_\infty.$$

Thus,

$$\|A\|_\infty \leq \max_{1 \leq i \leq n} \left(\sum_{j=1}^n |a_{ij}| \right), \quad (3.3)$$

since

$$\frac{\|Ax\|_\infty}{\|x\|_\infty} \leq \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|.$$

Now let k be the index of the row which has the maximum absolute row sum, i.e.,

$$\sum_{j=1}^n |a_{kj}| = \max_{1 \leq i \leq n} \left(\sum_{j=1}^n |a_{ij}| \right),$$

and let y be defined by

$$y_j = \begin{cases} \bar{a}_{kj}/|a_{kj}| & \text{if } a_{kj} \neq 0, \\ 0 & \text{if } a_{kj} = 0. \end{cases}$$

Hence, $\|y\|_\infty = 1$ (if $A \neq 0$). Also,

$$\begin{aligned} \|Ay\|_\infty &= \max_i \left| \sum_{j=1}^n a_{ij} y_j \right| \\ &\geq \left| \sum_{j=1}^n a_{kj} y_j \right| = \sum_{j=1}^n |a_{kj}| = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|. \end{aligned} \quad (3.4)$$

Since, from the definition of $\|A\|_\infty$, $\|A\|_\infty \geq \|Ay\|_\infty > 0$, (3.4) implies

$$\|A\|_\infty \geq \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|. \quad (3.5)$$

Combining (3.3) and (3.5) gives (a).

To prove (b), consider

$$\|Ax\|_1 = \sum_{i=1}^n \left| \sum_{j=1}^n a_{ij}x_j \right| \leq \sum_{i=1}^n \sum_{j=1}^n |a_{ij}|x_j \leq \max_{1 \leq j \leq n} \sum_{i=1}^n |a_{ij}| \|x\|_1.$$

(Recall $\|x\|_1 = \sum_{j=1}^n |x_j|$.) Thus,

$$\|A\|_1 \leq \max_{1 \leq j \leq n} \left(\sum_{i=1}^n |a_{ij}| \right). \quad (3.6)$$

Let k be such that $\sum_i |a_{ik}| = \max_j \left(\sum_i |a_{ij}| \right)$ and let $y = e_k$, where $(e_k)_j = \delta_{kj}$.

Then $\|y\|_1 = 1$, and

$$\|Ay\|_1 = \sum_{i=1}^n \left| \sum_{j=1}^n a_{ij}y_j \right| = \sum_{i=1}^n \left| \sum_{j=1}^n a_{ij}\delta_{jk} \right| = \sum_{i=1}^n |a_{ik}| = \max_{1 \leq j \leq n} \left(\sum_{i=1}^n |a_{ij}| \right).$$

However, from the definition of $\|A\|_1$, $\|A\|_1 \geq \|Ay\|_1$. Thus,

$$\|A\|_1 \geq \max_j \left(\sum_{i=1}^n |a_{ij}| \right). \quad (3.7)$$

Hence, (b) follows from (3.7) and (3.6). $\square$

Now consider $\|A\|_2$. Recall that $\|A\|_2 \neq \|A\|_E$. Also, recall the following.

- (i) $\rho(A) = \max_i |\lambda_i(A)| = \{\text{spectral radius of matrix } A\}$ (the eigenvalue of A with the largest magnitude).
- (ii) A Hermitian matrix B ($B^H = B$) has real eigenvalues and a linearly independent set of eigenvectors which are orthonormal with respect to the inner product $(\cdot, \cdot)$, where $(x, y) = y^H x$.

With this, we have:

PROPOSITION 3.5

$$\|A\|_2 = \sqrt{\rho(A^H A)}.$$

PROOF Let $x \neq 0$ in $\mathbb{C}^n$. Then

$$\frac{\|Ax\|_2^2}{\|x\|_2^2} = \frac{(Ax, Ax)}{(x, x)} = \frac{(Ax)^H Ax}{(x, x)} = \frac{x^H A^H Ax}{(x, x)} = \frac{(A^H Ax, x)}{(x, x)}.$$

Thus,

$$\frac{\|Ax\|_2}{\|x\|_2} = \left(\frac{(A^H Ax, x)}{(x, x)} \right)^{\frac{1}{2}}.$$

Now $A^H A$ is Hermitian. Let $\nu_i, i = 1, 2, \dots, n$, be the orthonormal eigenvectors of $A^H A$ and let

$$x = \sum_{i=1}^n c_i \nu_i.$$

Since $(\nu_i, \nu_j) = \delta_{ij}$ and since

$$A^H Ax = \sum_{i=1}^n \lambda_i c_i \nu_i,$$

it follows that

$$(x, x) = \sum_{i=1}^n |c_i|^2$$

and

$$(Ax, Ax) = (A^H Ax, x) = \sum_{i=1}^n \lambda_i |c_i|^2,$$

where $\lambda_i, 1 \leq i \leq n$, are the eigenvalues of $A^H A$. Now $\lambda_i = (A\nu_i, A\nu_i) \geq 0$, so

$$\frac{\|Ax\|_2}{\|x\|_2} = \left(\frac{\sum_{i=1}^n \lambda_i |c_i|^2}{\sum_{i=1}^n |c_i|^2} \right)^{\frac{1}{2}} \leq (\max_i \lambda_i)^{\frac{1}{2}} = \sqrt{\rho(A^H A)}. \quad (3.8)$$

Thus, $\|A\|_2 \leq \sqrt{\rho(A^H A)}$. Also, letting $x = \nu_k$ where $\lambda_k = \max_i \lambda_i = \rho(A^H A)$, it follows that

$$\|A\|_2 \geq \frac{\|A\nu_k\|_2}{\|\nu_k\|_2} = (A^H A\nu_k, \nu_k)^{\frac{1}{2}} = \lambda_k^{\frac{1}{2}} = \sqrt{\rho(A^H A)}. \quad (3.9)$$

Combining the inequalities (3.8) and (3.9), we obtain $\|A\|_2 = \sqrt{\rho(A^H A)}$. $\square$

It is interesting in view of Proposition 3.5 to revisit Proposition 3.3, namely, that $\rho(A) \leq \|A\|$ for any square matrix A and any matrix norm. An alternate proof of this proposition for induced matrix norms can be based on quotients $\|Ax\|/\|x\|$, where x is an arbitrary eigenvector of A . It is also interesting

to note that the difference between $\|A\|$ and $\rho(A)$ may be arbitrarily large. Consider

$$A = \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix}.$$

Then $\rho(A) = 0$ but $\|A\|_1 = 2$. However, the following result shows that there is a matrix norm arbitrarily close to $\rho(A)$.

PROPOSITION 3.6

Let A be a given $n \times n$ matrix and let $\epsilon > 0$. Then there exists an induced matrix norm $\|\cdot\|$ such that $\|A\| \leq \rho(A) + \epsilon$.

PROOF The proof rests on the result in linear algebra that any matrix is similar to an upper triangular matrix, i.e., given any matrix A , there exists a nonsingular matrix P such that $PAP^{-1} = T = \Lambda + U$ where T is upper triangular. Furthermore, the diagonal entries of T are the eigenvalues of A , i.e., if $\Lambda = \text{diag} [\lambda_1, \lambda_2, \dots, \lambda_n]$ then U has zero's on and below the diagonal. (This result is known as Schur's Theorem, and the decomposition $A = P^{-1}TP$ is known as the *Schur decomposition*.) With $\delta > 0$, define

$$D^{-1} = \text{diag}[1, \delta, \delta^2, \dots, \delta^{n-1}].$$

Then $C = DTD^{-1} = \Lambda + E$ where $E = (e_{ij}) = DUD^{-1}$ has elements

$$e_{ij} = \begin{cases} 0 & \text{if } j \leq i, \\ u_{ij}\delta^{j-i} & \text{if } j > i. \end{cases}$$

Hence, the elements of E can be made arbitrarily small by choosing δ small enough. Also, note that $A = P^{-1}D^{-1}CDP$. Since DP is nonsingular, a vector norm can be defined by

$$\|x\| = \|DPx\|_2 = (x^H P^H D^H D P x)^{\frac{1}{2}}.$$

The matrix norm induced by this vector norm is

$$\|A\| = \sup_{\|y\|=1} \|Ay\|.$$

For the particular A above,

$$\|Ay\| = \|DPAy\|_2 = \|CDPy\|_2.$$

Letting $z = DPy$, we have $\|Ay\| = \|Cz\|_2 = (z^H C^H C z)^{\frac{1}{2}}$. Put

$$C^H C = (\Lambda^H + E^H)(\Lambda + E) = \Lambda^H \Lambda + M(\delta),$$

where $M(\delta)$ is an $n \times n$ matrix whose elements are order δ , i.e., $|m_{ij}/\delta| \leq k$ for δ sufficiently small. Thus,

$$\|Ay\| = z^H C^H C z \leq \max_i |\lambda_i^2(A)| z^H z + |z^H M(\delta) z| \leq (\rho^2(A) + \mathcal{O}(\delta)) z^H z.$$

Since $z = DPy$ and $\|y\| = 1$, we have $\|y\| = \|DPy\|_2 = \|z\|_2 = 1$, that is, $z^H z = 1$. Therefore,

$$\|A\| \leq [\rho^2(A) + \mathcal{O}(\delta)]^{\frac{1}{2}} \leq \rho(A) + (\mathcal{O}(\delta))^{\frac{1}{2}}.$$

For δ sufficiently small, we can make $(\mathcal{O}(\delta))^{\frac{1}{2}} < \epsilon$. □

The following is now worth noting.

PROPOSITION 3.7

All matrix norms are equivalent, that is, given any two matrix norms $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$ there exist positive constants c_1 and c_2 such that

$$c_1 \|A\|_\alpha \leq \|A\|_\beta \leq c_2 \|A\|_\alpha.$$

PROOF By particular ordering of the elements of A , A can be viewed as a vector in $\mathbb{C}^{n^2}$. Thus, a matrix norm of A can be viewed as a vector norm in $\mathbb{C}^{n^2}$. Since any two vector norms are equivalent, any two matrix norms are equivalent. □

Note: Propositions 3.3, 3.6, and 3.7 give $\rho(A) \leq \|A\| \leq c\rho(A) + \epsilon$ where $\|\cdot\|$ is any matrix norm, c is a positive constant that depends on the norm $\|\cdot\|$ and matrix A , and $\rho(A)$ is the spectral radius of A . However, given A and $\epsilon > 0$, there is a norm $\|\cdot\|'$ such that $\rho(A) \leq \|A\|' \leq \rho(A) + \epsilon$.

We now consider sequences of matrices.

DEFINITION 3.27 $\{A_k\}_{k=1}^\infty$ converges to matrix A if and only if $\|A_k - A\| \rightarrow 0$ as $k \rightarrow \infty$ for some matrix norm.

Note: The choice of norm is not important from a theoretical point of view, since all matrix norms are equivalent. Thus, one should use the norm most suitable for the particular problem.

Note: It is clear that if $\|A_k - A\| \rightarrow 0$ as $k \rightarrow \infty$, then $a_{ij}^{(k)} \rightarrow a_{ij}$ as $k \rightarrow \infty$ for each i, j .

We now consider the sequence $\{A^k\}_{k=0}^\infty$, i.e., elements of the sequence are powers of A . We have the following convergence result:

THEOREM 3.5

Given an $n \times n$ matrix A , the following are equivalent.

$$(a) \lim_{k \rightarrow \infty} A^k = 0 \quad (\text{i.e., } \lim_{k \rightarrow \infty} \|A^k\| = 0).$$

(b) $\lim_{k \rightarrow \infty} A^k x = 0, \forall x \in \mathbb{C}^n$ (i.e., $\lim_{k \rightarrow \infty} \|A^k x\| = 0$).

(c) $\rho(A) < 1$.

(d) there exists a norm $\|\cdot\|$ such that $\|A\| < 1$.

REMARK 3.22 A matrix for which (a) holds is called a convergent matrix. $\square$

PROOF We will prove that (a) $\Rightarrow$ (b) $\Rightarrow$ (c) $\Rightarrow$ (d) $\Rightarrow$ (a).

(a) $\Rightarrow$ (b): We have $\|A^k x\| \leq \|A^k\| \|x\|$. Thus, $\lim_{k \rightarrow \infty} \|A^k x\| = 0$.

(b) $\Rightarrow$ (c): Let v be an eigenvector of A with associated eigenvalue λ . By (b), $\lim_{k \rightarrow \infty} \|A^k v\| = \lim_{k \rightarrow \infty} |\lambda|^k \|v\| = 0$. This implies that $|\lambda| < 1$. Since this holds for any eigenvalue of A , $\rho(A) < 1$.

(c) $\Rightarrow$ (d): we choose for $\rho(A) < 1, \epsilon$ such that $0 < \epsilon < 1 - \rho(A)$. Then, by Proposition 3.6, $\|A\| \leq \rho(A) + \epsilon < 1$.

(d) $\Rightarrow$ (a): we have $\|A^k\| \leq \|A\|^k$. Thus, $\lim_{k \rightarrow \infty} \|A^k\| \leq \lim_{k \rightarrow \infty} \|A\|^k = 0$, since $\|A\| < 1$. $\square$

REMARK 3.23 In condition (d), there exists should be stressed. Consider

$$A = \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix}.$$

$\|A\|_\infty = \|A\|_1 = \|A\|_2 = \|A\|_E = 2 > 1$, but $\lambda_1(A) = \lambda_2(A) = 0$, so $\rho(A) = 0$. Thus, $\lim_{k \rightarrow \infty} A^k = 0$. In fact, $A^k = 0$ for $k \geq 2$. $\square$

Two more useful results are now given.

PROPOSITION 3.8

The geometric series

$$\sum_{k=0}^{\infty} A^k = I + A + A^2 + A^3 + \cdots$$

converges to a certain matrix if and only if $A^k \rightarrow 0$ as $k \rightarrow \infty$. Furthermore, if $A^k \rightarrow 0$ as $k \rightarrow \infty$, then $I - A$ is nonsingular and

$$(I - A)^{-1} = \sum_{k=0}^{\infty} A^k.$$

PROOF Let $A^k \rightarrow 0$. By Theorem 3.5, $\rho(A) < 1$. Consider $B = I - A$. Then $\lambda(B) = 1 - \lambda(A)$, where $\lambda(B)$ is any eigenvalue of B and $\lambda(A)$ is the corresponding eigenvalue of A . Thus, $\lambda(B) \neq 0$ since $|\lambda(A)| < 1$. Hence, $I - A$ is nonsingular because no eigenvalues are zero. Now consider the identity

$$(I - A)(I + A + \cdots + A^k) = I - A^{k+1}.$$

This implies

$$(I + A + \cdots + A^k) = (I - A)^{-1}(I - A^{k+1}),$$

that is,

$$\lim_{k \rightarrow \infty} \sum_{j=0}^k A^j = (I - A)^{-1} \lim_{k \rightarrow \infty} (I - A^{k+1}) = (I - A)^{-1}.$$

Thus,

$$(I - A)^{-1} = \sum_{j=0}^{\infty} A^j.$$

Conversely, let $\sum_{k=0}^{\infty} A^k$ be a convergent series. Then

$$A^k = \sum_{j=0}^k A^j - \sum_{j=0}^{k-1} A^j \rightarrow 0$$

as $k \rightarrow \infty$. That is, a necessary condition for convergence of the series is $\lim_{k \rightarrow \infty} A^k = 0$. $\square$

PROPOSITION 3.9

Let $\|\cdot\|$ be a vector norm and $\|A\|$ its induced matrix norm, i.e.,

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}.$$

If $\|A\| < 1$, then $(I - A)$ is nonsingular and

$$\frac{1}{1 + \|A\|} \leq \|(I - A)^{-1}\| \leq \frac{1}{1 - \|A\|}.$$

PROOF By Theorem 3.5 and Proposition 3.8, $(I - A)$ is nonsingular. We also have $\|I\| = 1$, since $\|\cdot\|$ is an induced matrix norm. Thus,

$$1 = \|I\| = \|(I - A)^{-1}(I - A)\| \leq \|(I - A)^{-1}\| \|I - A\| \leq \|(I - A)^{-1}\| (1 + \|A\|).$$

Hence, $1/(1 + \|A\|) \leq \|(I - A)^{-1}\|$.

Also, $(I - A)^{-1} = I + A(I - A)^{-1}$. Thus,

$$\|(I - A)^{-1}\| \leq 1 + \|A\| \|(I - A)^{-1}\|,$$

that is,

$$(1 - \|A\|) \|(I - A)^{-1}\| \leq 1.$$

Hence, $\|(I - A)^{-1}\| \leq 1/(1 - \|A\|)$. $\square$

Note: This concludes, for now, the review of linear algebra. More results, such as for eigenvalues and eigenvectors and for irreducible matrices, will be given later in this chapter or in Chapter 5.

3.3 Direct Methods for Solving Linear Systems

We discuss elimination and factorization methods for solving linear systems in this section. We begin with Gaussian elimination.

3.3.1 Gaussian Elimination

We first present our notation.

3.3.1.1 Statement of the Problem

Given a nonsingular complex $n \times n$ matrix A and $b \in \mathbb{C}^n$, find $x \in \mathbb{C}^n$ such that $Ax = b$. That is, if $A = (a_{ij})$ and $b = (b_1, \dots, b_n)^T$ find $x = (x_1, x_2, \dots, x_n)^T$ such that

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1, \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2, \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n &= b_n. \end{aligned}$$

For a general matrix A , the goal of Gaussian elimination is to reduce A to an upper triangular matrix through a sequence of elementary row operations.⁵ We will see that this is equivalent to finding an invertible matrix M such that $MA = U$, where U is upper triangular, so $MAx = Mb$, that is, $Ux = Mb$.

⁵Recall that row operations involve interchanging two rows and replacement of a row by the sum of that row and a scalar multiple of another row; see Definition 3.12 on page 88.

Suppose that

$$U = \begin{pmatrix} a_{11}^{(n)} & a_{12}^{(n)} & \cdots & a_{1n}^{(n)} \\ 0 & a_{22}^{(n)} & \cdots & a_{2n}^{(n)} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{nn}^{(n)} \end{pmatrix} \quad \text{and} \quad Mb = \begin{pmatrix} b_1^{(n)} \\ b_2^{(n)} \\ \vdots \\ b_n^{(n)} \end{pmatrix}.$$

(The use of the index (n) will become clear shortly.) Then $Ux = Mb$ can be rapidly solved by a process called *backsolving* or *back substitution*. Specifically,

$$\begin{cases} x_n = b_n^{(n)} / a_{nn}^{(n)}, \\ x_k = \left(b_k^{(n)} - \sum_{j=k+1}^n a_{kj}^{(n)} x_j \right) / a_{kk}^{(n)} \quad \text{for } k = n-1, n-2, \dots, 1. \end{cases}$$

3.3.1.2 The Gaussian Elimination Procedure

Recall that we are trying to solve $Ax = b$, where A is nonsingular. For uniformity of notation, let $A = A^{(1)} = (a_{ij}^{(1)})$ and $b = b^{(1)} = (b_1^{(1)}, b_2^{(1)}, \dots, b_n^{(1)})^T$, so that $A^{(1)}x = b^{(1)}$.

Step 1: Assume that $a_{11}^{(1)} \neq 0$. (Otherwise, the nonsingularity of A guarantees that the rows of A can be interchanged in such a way that the new $a_{11}^{(1)}$ is nonzero.) Let

$$m_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}}, \quad 2 \leq i \leq n.$$

Now multiply the first equation of $A^{(1)}x = b^{(1)}$ by m_{i1} and subtract the result from the i -th equation. Repeat this for each i , $2 \leq i \leq n$. As a result, we obtain $A^{(2)}x = b^{(2)}$, where

$$A^{(2)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(2)} & \cdots & a_{nn}^{(2)} \end{pmatrix} \quad \text{and} \quad b^{(2)} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{pmatrix}.$$

$A^{(2)}$ is invertible, since row operations do not affect $|\det(A)|$, i.e.,

$$|\det(A^{(2)})| = |\det(A^{(1)})|.$$

Step 2: We consider the $(n-1) \times (n-1)$ submatrix $\tilde{A}^{(2)}$ of $A^{(2)}$ defined by $\tilde{A}^{(2)} = a_{ij}^{(2)}$, $2 \leq i, j \leq n$. We eliminate the first column of $\tilde{A}^{(2)}$

in a manner identical to the procedure for $A^{(1)}$. The result is system $A^{(3)}x = b^{(3)}$ where $A^{(3)}$ has the form

$$A^{(3)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & 0 & a_{33}^{(3)} & \cdots & a_{3n}^{(3)} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & a_{n3}^{(3)} & \cdots & a_{nn}^{(3)} \end{pmatrix}.$$

Steps 3 to $n - 1$: The process continues as above, where at the k -th stage we have $A^{(k)}x = b^{(k)}$, $1 \leq k \leq n - 1$, where

$$A^{(r)} = \begin{pmatrix} a_{11}^{(1)} & \cdots & & & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & & a_{2n}^{(2)} \\ \vdots & 0 & a_{33} & \cdots & \vdots \\ & \vdots & \ddots & \ddots & \\ & & & 0 & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ & & & & \vdots & & \vdots \\ 0 & \cdots & & 0 & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} \end{pmatrix} \quad \text{and } b^{(r)} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_{k-1}^{(k-1)} \\ b_k^{(k)} \\ \vdots \\ b_n^{(k)} \end{pmatrix}. \quad (3.10)$$

For every i , $k + 1 \leq i \leq n$, the k -th equation is multiplied by

$$m_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}$$

and subtracted from the i -th equation. (We assume, if necessary, a row is interchanged so that $a_{kk}^{(k)} \neq 0$.) After step $k = n - 1$, the resulting system is $A^{(n)}x = b^{(n)}$ where $A^{(n)}$ is upper triangular.

On a computer, this algorithm can be programmed as:

ALGORITHM 3.1

(Gaussian elimination, forward phase)

INPUT: $A \in L(\mathbb{R}^n)$ and $b \in \mathbb{R}^n$.

OUTPUT: $A^{(n)}$ and $b^{(n)}$.

FOR $k = 1, 2, \dots, n - 1$

FOR $i = k + 1, \dots, n$

(a) $m_{ik} \leftarrow a_{ik} / a_{kk}$.

(b) FOR $j = k, k + 1, \dots, n$

$$a_{ij} \leftarrow a_{ij} - m_{ik}a_{kj}.$$

END FOR

$$(c) \ b_i \leftarrow b_i - m_{ik}b_k.$$

END FOR

END FOR

END ALGORITHM 3.1.

Note: In Step (b) of Algorithm 3.1, we need only do the loop for $j = k + 1$ to n , since we know that the resulting $a_{k+1,k}$ will equal 0.

Back solving can be programmed as:

ALGORITHM 3.2

(Gaussian elimination, back solving phase)

INPUT: $A^{(n)}$ and $b^{(n)}$ from Algorithm 3.1.

OUTPUT: $x \in \mathbb{R}^n$ as a solution to $Ax = b$.

$$1. \ x_n \leftarrow b_n/a_{nn}.$$

$$2. \text{ FOR } k = n - 1, n - 2, \dots, 1$$

$$x_k \leftarrow (b_k - \sum_{j=k+1}^n a_{kj}x_j)/a_{kk}.$$

END FOR

END ALGORITHM 3.2.

Note: To solve $Ax = b$ using Gaussian elimination requires $\frac{1}{3}n^3 + \mathcal{O}(n^2)$ multiplications and divisions. (See Exercise 18 on page 182.)

3.3.1.3 The LU decomposition

Now let's consider the LU decomposition (triangular decomposition). Assume first that no row interchanges are performed in Gaussian elimination. Let

$$M^{(1)} = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ -m_{21} & & & \\ -m_{31} & & & \\ \vdots & & & \\ -m_{n1} & & & \end{array} \middle| \begin{array}{c} \\ \\ \\ I_{n-1} \\ \end{array} \right),$$

where I_{n-1} is the $(n-1) \times (n-1)$ identity matrix, and where m_{i1} , $2 \leq i \leq n$ are defined in Gaussian elimination. Then it is straightforward to see that $A^{(2)} = M^{(1)}A^{(1)}$ and $b^{(2)} = M^{(1)}b^{(1)}$. Also, $\det(M^{(1)}) = 1$, so $\det(A^{(2)}) = \det(A^{(1)})$. At the r -th stage of the Gaussian elimination process,

$$M^{(r)} = r\text{-th row} \left(\begin{array}{c|c|c} I_{r-1} & 0 & 0 \dots 0 \\ \hline 0 & 1 & 0 \dots 0 \\ \hline 0 & -m_{r+1,r} & \\ \vdots & \vdots & \\ 0 & -m_{n,r} & I_{n-r} \end{array} \right), \quad (3.11)$$

and, thus, $\det M^{(r)} = 1$. ($M^{(r)}$ is nonsingular.) Also,

$$(M^{(r)})^{-1} = \left(\begin{array}{c|c|c} I_{r-1} & 0 & 0 \dots 0 \\ \hline 0 & 1 & 0 \dots 0 \\ \hline 0 & m_{r+1,r} & \\ \vdots & \vdots & \\ 0 & m_{n,r} & I_{n-r} \end{array} \right), \quad (3.12)$$

where m_{ir} , $r+1 \leq i \leq n$ are given in the Gaussian elimination process (Exercise 20 on page 182). Also, it is easy to see that $A^{(r+1)} = M^{(r)}A^{(r)}$ and $b^{(r+1)} = M^{(r)}b^{(r)}$. (Note: We are assuming here that $a_{rr}^{(r)} \neq 0$ and no row interchanges are required.) Collecting the above results, we obtain $A^{(n)}x = b^{(n)}$, where

$$A^{(n)} = M^{(n-1)}M^{(n-2)} \dots M^{(1)}A^{(1)} \quad \text{and} \quad b^{(n)} = M^{(n-1)}M^{(n-2)} \dots M^{(1)}b^{(1)}.$$

Recalling that $A^{(n)}$ is upper triangular and setting $A^{(n)} = U$, we have

$$A = (M^{(n-1)}M^{(n-2)} \dots M^{(1)})^{-1}U. \quad (3.13)$$

Note: The product of two lower triangular matrices is lower triangular and the inverse of a nonsingular lower triangular matrix is lower triangular. (Exercise 19 on page 182) Thus, $L = (M^{(1)})^{-1}(M^{(2)})^{-1} \dots (M^{(n-1)})^{-1}$ is lower triangular. Hence, $A = LU$, i.e., A is expressed as a product of lower and upper triangular matrices. The result is called the *LU decomposition* (also known as the *LU factorization*, *triangular factorization*, or *triangular decomposition* of A). The final matrices L and U are given by:

$$L = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ m_{21} & 1 & 0 & \cdots & 0 \\ m_{31} & m_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \\ m_{n1} & m_{n2} & \cdots & m_{n,n-1} & 1 \end{pmatrix} \quad \text{and} \quad U = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & a_{nn}^{(n)} \end{pmatrix}. \quad (3.14)$$

(See Exercise 22 on page 182.) This decomposition can be so formed when no row interchanges are required.⁶ Thus, the original problem $Ax = b$ is transformed into $LUx = b$. This can be readily solved by forward substitution followed by backward substitution. That is, we first solve $Ly = b$ for y , then solve $Ux = y$ for x . This is a relatively fast procedure for large n ($\mathcal{O}(n^2)$ as opposed to $\mathcal{O}(n^3)$ for initially factoring A) and is especially useful when solutions are required for many different b 's for the same matrix A .

REMARK 3.24 If $A = LU$, then

$$\det(A) = \det(L) \det(U) = \prod_{j=1}^n a_{jj}^{(j)}.$$

Hence, Gaussian elimination is an efficient procedure for computing $\det(A)$. (Using expansion by minors to compute the determinant requires $\mathcal{O}(n!)$ multiplications.) $\square$

REMARK 3.25 The inverse A^{-1} can be rapidly found by solving n systems $Ax_j = e_j$ where $(e_j)_i = \delta_{ij}$. If $A = LU$, we perform n pairs of forward and backward solves. Then, $A^{-1} = (x_1, x_2, \dots, x_n)$. (Note that $AA^{-1} = (Ax_1, Ax_2, \dots, Ax_n) = I$.) $\square$

Note: In some software, the multiplying factors, that is, the nonzero off-diagonal elements of L , are stored in the locations of corresponding entries of A that are made equal to zero, thus obviating the need for extra storage. Effectively, such software returns the elements of L and U in the same array that was used to store A .

3.3.1.4 Pivoting in Gaussian Elimination

We have two questions:

1. When can Gaussian elimination be performed without row interchanges?
2. If row interchanges are employed, can Gaussian elimination always be employed?

THEOREM 3.6

(Existence of an LU factorization) Assume that $n \times n$ matrix A is nonsingular. Then $A = LU$ if and only if all the leading principal submatrices of A are

⁶When row interchanges are required, we need to insert a permutation matrix into the matrix product. We will define permutation matrix and discuss this later.

nonsingular.⁷ Moreover, the LU decomposition is unique, where L has unit diagonal elements.

REMARK 3.26 Two important types of matrices that have nonsingular leading principal submatrices are symmetric positive definite and strictly diagonally dominant, i.e.,

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}|, \quad \text{for } i = 1, 2, \dots, n.$$

□

PROOF (of Theorem 3.6) First, suppose that all leading principal submatrices of A are nonsingular. If we can show that $a_{kk}^{(k)} \neq 0$ in the Gaussian elimination procedure for each k , $1 \leq k \leq n$, then we have proved that $A = LU$. We prove this by induction. For $k = 1$, this is just $a_{11} \neq 0$. Now suppose that $a_{ii}^{(i)} \neq 0$ for $i = 1, 2, \dots, k-1$ so we have $A^{(2)}, A^{(3)}, \dots, A^{(k)}$ and $M^{(1)}, M^{(2)}, \dots, M^{(k)}$. We can write

$$A^{(k)} = M^{(k-1)} M^{(k-2)} \dots M^{(1)} A^{(1)} \quad (3.15)$$

as

$$\begin{aligned} \begin{pmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ A_{21}^{(k)} & A_{22}^{(k)} \end{pmatrix} &= \begin{pmatrix} M_{11}^{(k-1)} & 0 \\ M_{21}^{(k-1)} & M_{22}^{(k-1)} \end{pmatrix} \begin{pmatrix} M_{11}^{(k-2)} & 0 \\ M_{21}^{(k-2)} & M_{22}^{(k-2)} \end{pmatrix} \\ &\quad \dots \begin{pmatrix} M_{11}^{(1)} & 0 \\ M_{21}^{(1)} & M_{22}^{(1)} \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \end{aligned}$$

where A_{11} is $k \times k$, A_{12} is $k \times (n-k)$, A_{21} is $(n-k) \times k$ and A_{22} is $(n-k) \times (n-k)$. In the above, $A_{11}^{(k)}$ is the leading principal submatrix of order k and matrices $M^{(k-1)}, M^{(k-2)}, \dots, M^{(1)}$ and A are partitioned accordingly. Since $M^{(i)}$ are lower triangular, it follows from (3.15) that $A_{11}^{(k)} = M_{11}^{(k-1)} \dots M_{11}^{(1)} A_{11}$. But all $M_{11}^{(i)}$ are nonsingular, and A_{11} is nonsingular by assumption. Hence, $A_{11}^{(k)}$

⁷Recall that the leading principal submatrices of A have the form

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{k1} & \dots & a_{kk} \end{pmatrix} \quad \text{for } k = 1, 2, \dots, n.$$

is nonsingular, so

$$\det(A_{11}^{(k)}) = \det \begin{pmatrix} a_{11}^{(1)} & \cdots & a_{1n}^{(1)} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & a_{kk}^{(k)} \end{pmatrix} = a_{11}^{(1)} a_{22}^{(2)} \cdots a_{kk}^{(k)} \neq 0,$$

so $a_{kk}^{(k)} \neq 0$.

For the converse, suppose that $A = LU$. We need to show that all leading principal submatrices are nonsingular. We write

$$A = LU = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix},$$

where A_{11} is $k \times k$ and L and U are partitioned accordingly. Since A is nonsingular, L and U are nonsingular and hence L_{11} and U_{11} are nonsingular. But then $A_{11} = L_{11}U_{11}$ is nonsingular.

Finally, consider uniqueness. Assume that $L_1U_1 = L_2U_2 = A$. Then $U_1U_2^{-1} = L_1^{-1}L_2$. But $L_1^{-1}L_2$ is lower triangular with diagonal elements unity, and $U_1U_2^{-1}$ is upper triangular. Thus, $U_1U_2^{-1} = L_1^{-1}L_2 = I$, so $L_1 = L_2$ and $U_1 = U_2$. $\square$

We now consider our second question, "If row interchanges are employed, can Gaussian elimination be performed for any nonsingular A ?" The following definition will be useful.

DEFINITION 3.28 *A permutation matrix P is a matrix whose columns consist of the n different vectors $e_j, 1 \leq j \leq n$, in any order.*

For example,

$$P = (e_1, e_3, e_4, e_2) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Since a permutation matrix is a matrix whose columns are a rearrangement of the columns of the identity matrix, the effect of multiplying by a permutation matrix P on the left is to rearrange the rows of A in the order in which the e_j^T appear in the rows of P . For example,

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \\ 1 & 2 & 3 \end{pmatrix}.$$

Thus, by proper choice of P , any two or more rows can be interchanged.

Note: $\det P = \pm 1$, since P is obtained from I by row interchanges.

Now, Gaussian elimination with row interchanges can be performed by the following matrix operations:⁸

$$A^{(n)} = M^{(n-1)}P^{(n-1)}M^{(n-2)}P^{(n-2)} \dots M^{(2)}P^{(2)}M^{(1)}P^{(1)}A.$$

$$b^{(n)} = M^{(n-1)}P^{(n-1)} \dots M^{(2)}P^{(2)}M^{(1)}P^{(1)}b.$$

It follows that $U = \hat{L}A^{(1)}$, where $\hat{L}$ is no longer lower triangular. However, if we perform all the row interchanges first, at once, then

$$M^{(n-1)} \dots M^{(1)}PAx = M^{(n-1)}M^{(n-2)} \dots M^{(1)}Pb,$$

or

$$\tilde{L}PAx = \tilde{L}Pb,$$

so

$$\tilde{L}PA = U.$$

Thus,

$$PA = \tilde{L}^{-1}U = LU.$$

We thus have the following result:

THEOREM 3.7

If A is a nonsingular $n \times n$ matrix, then there is a permutation matrix P such that $PA = LU$, where L is lower triangular and U is upper triangular. (Note: $\det(PA) = \pm \det(A) = \det(L)\det(U)$.)

We now examine the technique of pivoting in Gaussian elimination. Consider the system

$$0.0001x_1 + x_2 = 1$$

$$x_1 + x_2 = 2.$$

The exact solution of this system is $x_1 \approx 1.00010$ and $x_2 \approx 0.99990$. Let us solve the system using Gaussian elimination without row interchanges. We will assume calculations are performed using three-digit rounding decimal arithmetic. We obtain

$$m_{21} \leftarrow \frac{a_{21}^{(1)}}{a_{11}^{(1)}} \approx 0.1 \times 10^5,$$

$$a_{22}^{(2)} \leftarrow a_{22}^{(1)} - m_{21}a_{12}^{(1)} \approx 0.1 \times 10^1 - 0.1 \times 10^5 \approx -0.100 \times 10^5.$$

⁸When implementing Gaussian elimination, we usually don't actually multiply full n by n matrices together, since this is not efficient. However, viewing the process as matrix multiplications has advantages when we analyze it.

Also, $b^{(2)} \approx (0.1 \times 10^1, -0.1 \times 10^5)^T$, so the computed (approximate) upper triangular system is

$$\begin{aligned} 0.1 \times 10^{-3}x_1 + 0.1 \times 10^1x_2 &= 0.1 \times 10^1, \\ -0.1 \times 10^5x_2 &= -0.1 \times 10^5, \end{aligned}$$

whose solutions are $x_2 = 1$ and $x_1 = 0$. If instead, we first interchange the equations so that $a_{11}^{(1)} = 1$, we find that $x_1 = x_2 = 1$, correct to the accuracy used.

Note: Small values of $a_{rr}^{(r)}$ in the r -th stage lead to large values of the m_{ir} 's and may result in a loss of accuracy. Therefore, we want the pivots $a_{rr}^{(r)}$ to be large.

Two common pivoting strategies are:

Partial pivoting: In partial pivoting, the $a_{ir}^{(r)}$ for $r \leq i \leq n$, in the r -th column of $A^{(r)}$ is searched to find the element of largest absolute value, and row interchanges are made to place that element in the pivot position.

Full pivoting: In full pivoting, the pivot element is selected as the element $a_{ij}^{(r)}$, $r \leq i, j \leq n$ of maximum absolute value among all elements of the $(n-r) \times (n-r)$ submatrix of $A^{(r)}$. This strategy requires row and column interchanges.

Note: In practice, partial pivoting in most cases is adequate.

Note: For some classes of matrices, no pivoting strategy is required for a stable elimination procedure. For example, no pivoting is required for a real symmetric positive definite matrix or for a strictly diagonally dominant matrix [99].

We now present a formal algorithm for Gaussian elimination with partial pivoting. In reading this algorithm, recall that

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n. \end{aligned}$$

ALGORITHM 3.3

(Solution of a linear system of equations with Gaussian elimination with partial pivoting and back-substitution)

INPUT: $A \in L(\mathbb{R}^n)$ and $b \in \mathbb{R}^n$.

OUTPUT: An approximate solution⁹ x to $Ax = b$.

FOR $k = 1, 2, \dots, n-1$

1. Find ℓ such that $|a_{\ell k}| = \max_{k \leq j \leq n} |a_{jk}|$ ($k \leq \ell \leq n$).

2. Interchange row k with row ℓ

$$\left\{ \begin{array}{l} c_j \leftarrow a_{kj} \\ a_{kj} \leftarrow a_{\ell j} \\ a_{\ell j} \leftarrow c_j \end{array} \right\} \text{ for } j = 1, 2, \dots, n, \text{ and } \left\{ \begin{array}{l} d \leftarrow b_k \\ b_k \leftarrow b_\ell \\ b_\ell \leftarrow d \end{array} \right\}.$$

3. FOR $i = k+1, \dots, n$

(a) $m_{ik} \leftarrow a_{ik}/a_{kk}$.

(b) FOR $j = k+1, \dots, n$

$$a_{ij} \leftarrow a_{ij} - m_{ik}a_{kj}.$$

END FOR

(c) $b_i \leftarrow b_i - m_{ik}b_k$.

END FOR

4. Backsubstitution:

(a) $x_n \leftarrow b_n/a_{nn}$ and

(b) $x_k \leftarrow \left(b_k - \sum_{j=k+1}^n a_{kj}x_j \right) / a_{kk}$, for $k = n-1, n-2, \dots, 1$.

END FOR

END ALGORITHM 3.3.

REMARK 3.27 In Algorithm 3.3, the computations are arranged “serially,” that is, they are arranged so each individual addition and multiplication is done separately. However, it is efficient on modern machines, that have “pipelined” operations and usually also have more than one processor, to think of the operations as being done on vectors. Furthermore, we don’t necessarily need to change entire rows, but just keep track of a set of indices indicating which rows are interchanged; for large systems, this saves a significant number of storage and retrieval operations. For views of the Gaussian elimination process in terms of vector operations, see [34]. For an example of software that takes account of the way machines are built, see [4]. $\square$

⁹approximate because of roundoff error

REMARK 3.28 One can show that the final system is $A^{(n)}x = b^{(n)}$, where $A^{(n)}$ is upper triangular, $A^{(n)} = M^{(n-1)}P^{(n-1)} \dots M^{(1)}P^{(1)}A = MA$, and $b^{(n)} = M^{(n-1)}P^{(n-1)} \dots M^{(1)}P^{(1)}b$, where the first $r-1$ columns of $M^{(r)}$ are the first $r-1$ columns of the identity matrix, the last $n-r$ columns of $M^{(r)}$ are the last $n-r$ columns of the identity matrix, and the r -th column of $M^{(r)}$ is

$$M_{:,r}^{(r)} = (0_{1 \times r-1} \quad 1 \quad -m_{r+1,r} \quad \dots \quad -m_{n,r})^T,$$

where $0_{1 \times r-1}$ is $r-1$ zero's, and

$$P^{(r)} = (e_1, \dots, e_{r-1}, e_j, e_{r+1}, \dots, e_{j-1}, e_r, e_{j+1}, \dots, e_n)$$

is an $n \times n$ permutation matrix where, at the r -th step, row r is interchanged with row j , and where e_j denotes the j -th column of the identity matrix. $\square$

We have $A = M^{-1}A^{(n)} = M^{-1}U$, i.e., $A^{(n)} = U$ is upper triangular. However, unless $P^{(r)} = I$ for every r , M^{-1} is not lower triangular. (We have proved that given any nonsingular matrix A , there exists a nonsingular matrix M such that $MA = U$.)

REMARK 3.29 Note that

$$\begin{aligned} \det(A) &= \det(M^{-1}) \det(U) \\ &= \det(P^{(1)})^{-1} \det(M^{(1)})^{-1} \dots \det(P^{(n-1)})^{-1} \det(M^{(n-1)})^{-1} \det(U). \end{aligned}$$

But

$$\det(P^{(r)})^{-1} = \det(P^{(r)}) = \begin{cases} -1 & \text{if a row has been interchanged,} \\ 1 & \text{if no row has been interchanged,} \end{cases}$$

and $\det(M^{(r)})^{-1} = 1$. Thus,

$$\det(A) = (-1)^K \det(U) = (-1)^K a_{11}^{(1)} a_{22}^{(2)} \dots a_{nn}^{(n)},$$

where K is the number of row interchanges made. $\square$

REMARK 3.30 The inverse A^{-1} can be found by solving n systems $Ax_j = e_j$, $j = 1, 2, \dots, n$, where $(e_j)_k = \delta_{jk}$ in a simultaneous manner. Then,

$$A^{-1} = (x_1, x_2, \dots, x_n).$$

$\square$

We now consider some special but commonly encountered kinds of matrices.

3.3.2 Symmetric, Positive Definite Matrices

We first characterize positive definite matrices.

THEOREM 3.8

Let A be a real symmetric $n \times n$ matrix. Then A is positive definite if and only if there exists an invertible lower triangular matrix L such that $A = LL^T$. Furthermore, we can choose the diagonal elements of L , ℓ_{ii} , $1 \leq i \leq n$, to be positive numbers.

Note: The decomposition with positive ℓ_{ii} is called the *Cholesky factorization* of A . It can be shown that this decomposition is unique.

PROOF (of Theorem 3.8) Recall that A is positive definite if $x^T Ax \geq 0$ for all $x \in \mathbb{R}^n$, with equality only if $x = 0$.

Part 1 (factorization exists implies positive definite) Let $A = LL^T$ with L invertible. Then $x^T Ax = x^T LL^T x$. Let $y = L^T x$. Then $x^T Ax = y^T y = y_1^2 + y_2^2 + \cdots + y_n^2 \geq 0$, with equality only if $y = 0$. But since L^T is invertible, we have $y = 0$ only if $x = 0$. Therefore, A is positive definite.

Part 2 (positive definite implies factorization exists) Let A be symmetric positive definite. It can be shown that the principal minor δ_i satisfies [69]:

$$\delta_i = \det \begin{pmatrix} a_{11} & \cdots & a_{1i} \\ & \ddots & \\ a_{i1} & \cdots & a_{ii} \end{pmatrix} > 0$$

for $i = 1, 2, \dots, n$, for positive definite matrices. Thus, by Theorem 3.6, A has the LU decomposition

$$A = \hat{L}U = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ * & 1 & 0 & \cdots & 0 \\ * & * & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ * & * & * & \cdots & 1 \end{pmatrix} \begin{pmatrix} u_{11} & * & * & \cdots & * \\ 0 & u_{22} & * & \cdots & * \\ 0 & 0 & u_{33} & \cdots & * \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix}.$$

That is, $\hat{L}$ is lower triangular with unit diagonal entries and U is upper triangular. Since $\prod_{j=1}^i u_{jj} = \delta_i > 0$, $u_{ii} > 0$ for $1 \leq i \leq n$. Define the diagonal matrix $\Lambda = \text{diag}(\sqrt{u_{11}}, \sqrt{u_{22}}, \dots, \sqrt{u_{nn}})$. Then $A = \hat{L}U = \hat{L}\Lambda\Lambda^{-1}U$. Now

define $L = \hat{L}\Lambda$ and $V = \Lambda^{-1}U$. Then we have $A = LV$, with

$$L = \begin{pmatrix} \sqrt{u_{11}} & 0 & \cdots & 0 \\ * & \sqrt{u_{22}} & \ddots & \vdots \\ * & * & \ddots & 0 \\ * & * & * & \sqrt{u_{nn}} \end{pmatrix}, \quad V = \begin{pmatrix} \sqrt{u_{11}} & * & * & * \\ 0 & \sqrt{u_{22}} & * & * \\ \vdots & \ddots & \ddots & * \\ 0 & \cdots & 0 & \sqrt{u_{nn}} \end{pmatrix}.$$

But $A = A^T$, so $LV = (LV)^T = V^T L^T$, and, since L is invertible, we have $V(L^T)^{-1} = L^{-1}V^T$. Since $(L^T)^{-1}$ is upper triangular and V^T is lower triangular, $V(L^T)^{-1}$ is upper triangular with unit elements along the diagonal. Similarly, $L^{-1}V^T$ is lower triangular with unit elements along the diagonal. Thus, $V(L^T)^{-1} = L^{-1}V^T = I$, so $V = L^T$, and thus $A = LL^T$. $\square$

REMARK 3.31 L can be computed using a variant of Gaussian elimination. Set $\ell_{11} = \sqrt{a_{11}}$ and $\ell_{j1} = a_{j1}/\sqrt{a_{11}}$ for $2 \leq j \leq n$. (Note that $x^T A x > 0$, and the choice $x = e_j$ implies that $a_{jj} > 0$.) Then, for $i = 1, 2, 3, \dots, n$, set

$$\ell_{ii} = \left[a_{ii} - \sum_{k=1}^{i-1} (\ell_{ik})^2 \right]^{\frac{1}{2}}$$

$$\ell_{ji} = \frac{1}{\ell_{ii}} \left[a_{ji} - \sum_{k=1}^{i-1} \ell_{ik} \ell_{jk} \right] \text{ for } i+1 \leq j \leq n.$$

$\square$

Note: If A is real symmetric and L can be computed using the previous note, then A is positive definite. (This is an efficient way to show positive definiteness.)

Note: To solve $Ax = b$ where A is real symmetric positive definite, L can be formed using the first note and the pair $Ly = b$ and $L^T x = y$ can be solved for x .

Note: The multiplication and division count for Cholesky decomposition is

$$n^3/6 + \mathcal{O}(n^2).$$

Thus, for large n , about $1/2$ the multiplications and divisions are required compared to standard Gaussian elimination.

3.3.3 Tridiagonal Matrices

A tridiagonal matrix is a matrix of the form

$$A = \begin{pmatrix} a_1 & c_1 & 0 & \cdots & & 0 \\ b_2 & a_2 & c_2 & 0 & \cdots & 0 \\ 0 & b_3 & a_3 & c_3 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & b_{n-1} & a_{n-1} & c_{n-1} \\ 0 & \cdots & & 0 & b_n & a_n \end{pmatrix}.$$

Suppose that A can be decomposed into a product of two bidiagonal matrices, that is,

$$A = LU = \begin{pmatrix} \alpha_1 & 0 & \cdots & 0 \\ b_2 & \alpha_2 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & b_n & \alpha_n \end{pmatrix} \begin{pmatrix} 1 & \gamma_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ & & \gamma_{n-1} & \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \quad (3.16)$$

which gives

$$\begin{aligned} \alpha_1 &= a_1, \\ \gamma_1 &= c_1/\alpha_1, \\ \alpha_i &= a_i - b_i\gamma_{i-1} \quad \text{for } i = 2, 3, \dots, n, \\ \gamma_i &= c_i/\alpha_i \quad \text{for } i = 2, \dots, n-1. \end{aligned} \quad (3.17)$$

Therefore, if $\alpha_i \neq 0$, $1 \leq i \leq n$, we can compute the decomposition (3.16). Furthermore, we can compute the solution to $Ax = f = (f_1, f_2, \dots, f_n)^T$ by successively solving $Ly = f$ and $Ux = y$, i.e.,

$$\begin{aligned} y_1 &= f_1/\alpha_1, \\ y_i &= (f_i - b_i y_{i-1})/\alpha_i \quad \text{for } i = 2, 3, \dots, n, \\ x_n &= y_n, \\ x_j &= (y_j - \gamma_j x_{j+1}) \quad \text{for } j = n-1, n-2, \dots, 1. \end{aligned} \quad (3.18)$$

Sufficient conditions to guarantee decomposition (3.16) are given in the following theorem.

THEOREM 3.9

Suppose the elements a_i , b_i , and c_i of A satisfy $|a_1| > |c_1| > 0$, $|a_i| \geq |b_i| + |c_i|$, and $b_i c_i \neq 0$ for $2 \leq i \leq n-1$, and suppose $|a_n| > |b_n| > 0$. Then A is invertible and the α_i 's are nonzero. (Consequently, the factorization (3.16) is possible.)

PROOF Since $\alpha_1 = a_1$ and $\gamma_1 = c_1/\alpha_1$, we have $\alpha_1 \neq 0$ and $|\gamma_1| < 1$. We now proceed inductively. (Recall that $\alpha_i = a_i - b_i\gamma_{i-1}$, $\gamma_i = c_i/\alpha_i$.) Suppose

that, for some j such that $2 \leq j \leq n$, we have $|\gamma_{j-1}| < 1$. Then

$$|a_j - b_j \gamma_{j-1}| \geq |a_j| - |b_j| |\gamma_{j-1}| > |a_j| - |b_j| > 0,$$

since $|a_j| - |b_j| \geq |c_j| > 0$. Thus, $|\alpha_j| > 0$, whence $\alpha_j \neq 0$. Also,

$$|\gamma_j| = \frac{|c_j|}{|\alpha_j|} = \frac{|c_j|}{|a_j - b_j \gamma_{j-1}|} < \frac{|c_j|}{|a_j| - |b_j|} \leq 1,$$

since $|a_j| - |b_j| \geq |c_j|$. Thus, $|\gamma_j| < 1$. The inductive proof is thus finished. Hence, all α_i 's are nonzero. The nonsingularity of A follows from

$$\det(A) = \det(L) \det(U) = \prod_{j=1}^n \alpha_j \neq 0.$$

□

Note: It can be verified that solution of a linear system having tridiagonal coefficient matrix using (3.17) and (3.18) requires $(5n - 4)$ multiplications and divisions and $3(n - 1)$ additions and subtractions. (Recall that we need $n^3/3 + \mathcal{O}(n^2)$ multiplications and divisions for Gaussian elimination.) Storage requirements are also drastically reduced to $3n$ locations, versus n^2 for a full matrix.

3.3.4 Block Tridiagonal Matrices

We now consider briefly block tridiagonal matrices, that is, matrices of the form

$$A = \begin{pmatrix} A_1 & C_1 & 0 & \cdots & 0 & 0 \\ B_2 & A_2 & C_2 & 0 & \cdots & 0 \\ 0 & B_3 & A_3 & C_3 & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & B_{n-1} & A_{n-1} & C_{n-1} \\ 0 & 0 & 0 & 0 & B_n & A_n \end{pmatrix},$$

where A_i, B_i , and C_i are $m \times m$ matrices. Analogous to the tridiagonal case, we construct a factorization of the form

$$A = \begin{pmatrix} \hat{A}_1 & 0 & \cdots & 0 \\ B_2 & \hat{A}_2 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ 0 & \cdots & B_n & \hat{A}_n \end{pmatrix} \begin{pmatrix} I & E_1 & \cdots & 0 \\ 0 & I & \ddots & 0 \\ 0 & \ddots & \ddots & E_{n-1} \\ 0 & \cdots & 0 & I \end{pmatrix}.$$

Provided the $\hat{A}_i$, $1 \leq i \leq n$, are nonsingular, we can compute:

$$\begin{aligned}\hat{A}_1 &= A_1, \\ E_1 &= \hat{A}_1^{-1}C_1, \\ \hat{A}_i &= A_i - B_i E_{i-1} \quad \text{for } 2 \leq i \leq n, \\ E_i &= \hat{A}_i^{-1}C_i, \quad \text{for } 2 \leq i \leq n-1.\end{aligned}$$

For efficiency, the $\hat{A}_i^{-1}$ are generally not computed, but instead, the columns of E_i are computed by factoring $\hat{A}_i$ and solving a pair of triangular systems. That is, $\hat{A}_i E_i = C_i$ with $\hat{A}_i = L_i U_i$ becomes $L_i U_i E_i = C_i$.

PROPOSITION 3.10

If the inverses $\hat{A}_i^{-1}$ are computed, the total number of multiplications and divisions to complete the factorization into a lower block triangular and an upper block triangular matrix as just described is $3(n-1)m^3$ (depending on the algorithm used to compute inverses), while, if the E_i 's are computed using Gaussian elimination, the leading term $3nm^3$ is reduced to $\frac{4}{3}nm^3$.

PROOF The proof is left as Exercise 24 on page 182. □

Since A is an $nm \times nm$ matrix, standard Gaussian elimination would require $\frac{1}{3}n^3m^3 + \mathcal{O}(n^2m^2)$ multiplications and divisions. Clearly, tremendous savings are achieved by taking advantage of the zero elements.

Now consider

$$Ax = b, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix},$$

where $x_i, b_i \in \mathbb{R}^m$. Then, with the factorization $A = LU$, $Ax = b$ can be solved as follows: $Ly = b$, $Ux = y$, with

$$\begin{aligned}\hat{A}_1 y_1 &= b_1, \\ \hat{A}_i y_i &= (b_i - B_i y_{i-1}) \quad \text{for } i = 2, \dots, n, \\ x_n &= y_n, \\ x_j &= y_j - E_j x_{j+1} \quad \text{for } j = n-1, \dots, 1.\end{aligned}$$

3.3.5 Roundoff Error and Conditioning in Gaussian Elimination

Round-off error analysis for Gaussian elimination can be divided into a description of condition numbers and ill-conditioned matrices, round-off error analysis, and iterative refinement.

3.3.5.1 Condition Numbers

We begin with the following:

DEFINITION 3.29 *If the solution x of $Ax = b$ changes drastically when A or b is perturbed slightly, then the system $Ax = b$ is called ill-conditioned.*

Note: Because rounding errors are unavoidable with floating point arithmetic, much accuracy can be lost during Gaussian elimination for ill-conditioned systems. In fact, the final solution may be considerably different than the exact solution.

Example 3.7

An ill-conditioned system is

$$Ax = \begin{pmatrix} 1 & 0.99 \\ 0.99 & 0.98 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1.99 \\ 1.97 \end{pmatrix}, \text{ whose exact solution is } x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

However,

$$Ax = \begin{pmatrix} 1.989903 \\ 1.970106 \end{pmatrix} \quad \text{has solution } x = \begin{pmatrix} 3 \\ -1.0203 \end{pmatrix}.$$

Thus, a change of

$$\delta b = \begin{pmatrix} -0.000097 \\ 0.000106 \end{pmatrix} \quad \text{produces a change } \delta x = \begin{pmatrix} 2.000 \\ -2.0203 \end{pmatrix}.$$

□

We first study the phenomenon of ill-conditioning, then study roundoff error in Gaussian elimination. We begin with

THEOREM 3.10

Let $\|\cdot\|_\beta$ be an induced matrix norm. Let x be the solution of $Ax = b$ with A an $n \times n$ invertible complex matrix. Let $x + \delta x$ be the solution of

$$(A + \delta A)(x + \delta x) = b + \delta b. \quad (3.19)$$

Assume that

$$\|\delta A\|_\beta \|A^{-1}\|_\beta < 1. \quad (3.20)$$

Then

$$\frac{\|\delta x\|_\beta}{\|x\|_\beta} \leq \kappa_\beta(A)(1 - \|\delta A\|_\beta \|A^{-1}\|_\beta)^{-1} \left(\frac{\|\delta b\|_\beta}{\|b\|_\beta} + \frac{\|\delta A\|_\beta}{\|A\|_\beta} \right), \quad (3.21)$$

where

$$\kappa_\beta(A) = \|A\|_\beta \|A^{-1}\|_\beta$$

is defined to be the condition number of the matrix A with respect to norm $\|\cdot\|_\beta$.

PROOF By subtracting $Ax = b$ from (3.19), we obtain

$$(A + \delta A)\delta x = \delta b - \delta Ax.$$

Then (3.20) and Proposition 3.9 (on page 104) imply the following.

(1) $(I + A^{-1}\delta A)$ is invertible.

(2) Since $A + \delta A = A(I + A^{-1}\delta A)$, we have $(A + \delta A)^{-1}$ exists.

Thus, by (1) and (2) above,

$$\delta x = (A + \delta A)^{-1}(\delta b - \delta Ax) = (I + A^{-1}\delta A)^{-1}A^{-1}(\delta b - \delta Ax).$$

Hence,

$$\begin{aligned} \|\delta x\|_\beta &\leq \|(I + A^{-1}\delta A)^{-1}\|_\beta \|A^{-1}\|_\beta \|\delta b - \delta Ax\|_\beta \\ &\leq \frac{1}{1 - \|A^{-1}\|_\beta \|\delta A\|_\beta} \|A^{-1}\|_\beta (\|\delta b\|_\beta + \|\delta A\|_\beta \|x\|_\beta), \end{aligned}$$

where the last inequality follows from Proposition 3.9 on page 104. Hence,

$$\frac{\|\delta x\|_\beta}{\|x\|_\beta} \leq \frac{\kappa_\beta(A)}{1 - \|A^{-1}\|_\beta \|\delta A\|_\beta} \left(\frac{\|\delta b\|_\beta}{\|A\|_\beta \|x\|_\beta} + \frac{\|\delta A\|_\beta}{\|A\|_\beta} \right),$$

and (3.21) follows from $\|b\|_\beta = \|Ax\|_\beta \leq \|A\|_\beta \|x\|_\beta$. □

Note: The condition number $\kappa_\beta(A) \geq 1$ for any induced matrix norm and any matrix A , since

$$1 = \|I\|_\beta = \|A^{-1}A\|_\beta \leq \|A^{-1}\|_\beta \|A\|_\beta = \kappa_\beta(A).$$

Note: If $\delta A = 0$, we have

$$\frac{\|\delta x\|_\beta}{\|x\|_\beta} \leq \kappa_\beta(A) \frac{\|\delta b\|_\beta}{\|b\|_\beta},$$

and if $\delta b = 0$, then

$$\frac{\|\delta x\|_\beta}{\|x\|_\beta} \leq \frac{\kappa_\beta(A)}{1 - \|A^{-1}\|_\beta \|\delta A\|_\beta} \frac{\|\delta A\|_\beta}{\|A\|_\beta}.$$

Note: There exist perturbations δx and δb for which (3.21) holds with equality. That is, inequality (3.21) is sharp.

REMARK 3.32 Consider $\|\cdot\|_\beta = \|\cdot\|_2$, where

$$\|A\|_2 = \sup_{x \neq 0} \left\{ \frac{\|Ax\|_2}{\|x\|_2} \right\}.$$

Let $\mu_1 \geq \mu_2 \geq \mu_3 \geq \cdots \geq \mu_n > 0$ be the eigenvalues of $A^H A$, with A invertible. (Recall that, when A is nonsingular, $A^H A$ is positive definite and has n positive eigenvalues.) Thus,

$$\|A\|_2 = \sqrt{\rho(A^H A)} = \sqrt{\mu_1},$$

and

$$\|A^{-1}\|_2 = \sqrt{\rho((A^{-1})^H A^{-1})} = \sqrt{\rho((A^H A)^{-1})} = \frac{1}{\sqrt{\mu_n}}.$$

(Notice that the eigenvalues of C^{-1} are $1/\lambda_i$, $1 \leq i \leq n$, if the eigenvalues of C are λ_i , $1 \leq i \leq n$.) Thus,

$$\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2 = \sqrt{\mu_1/\mu_n}.$$

In the particular case that A is Hermitian,

$$\kappa_2(A) = |\lambda|/|\sigma|,$$

where $|\lambda| = \max_{1 \leq i \leq n} |\lambda_i(A)|$ and $|\sigma| = \min_{1 \leq i \leq n} |\lambda_i(A)|$. □

Note: For a unitary matrix U , i.e., $U^H U = I$, we have $\kappa_2(U) = 1$ (Exercise 26 on page 182). Such a matrix is called *perfectly conditioned*, since $\kappa_\beta(A) \geq 1$ for any β and A .

Example 3.8

Consider the earlier example (Example 3.7) of an ill-conditioned matrix:

$$A = \begin{pmatrix} 1 & 0.99 \\ 0.99 & 0.98 \end{pmatrix}, \text{ with eigenvalues } \lambda_1 \approx 1.98005, \lambda_2 \approx -0.00005.$$

Since A is Hermitian, $\kappa_2(A) = |\lambda_1|/|\lambda_2| \approx 39601$. Recall that

$$A \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1.99 \\ 1.97 \end{pmatrix}, \quad x_1 = x_2 = 1,$$

and the perturbed system was

$$A \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{pmatrix} = \begin{pmatrix} 1.989903 \\ 1.970106 \end{pmatrix}, \quad \tilde{x}_1 = 3.000, \quad \tilde{x}_2 = -1.0203.$$

Thus, a change

$$\delta b = \begin{pmatrix} -0.000097 \\ 0.000106 \end{pmatrix} \quad \text{produced a change } \delta x = \begin{pmatrix} 2.0000 \\ -2.0203 \end{pmatrix}.$$

Computing $\|\delta x\|_2/\|x\|_2 \approx 2.010175$ and $\|\delta b\|_2/\|b\|_2 = 0.513123 \times 10^{-4}$, we have $\|\delta x\|_2/\|x\|_2 \approx 39175\|\delta b\|_2/\|b\|_2$. (Recall that $\kappa(A) = 39601$.) Thus, in this example, (3.21) is fairly sharp. (Note that $\delta A = 0$.) This example clearly illustrates the phenomenon of ill-conditioning. (An uncertainty of 10^{-4} in elements of b results in an uncertainty of about 2 in elements of x .) $\square$

A classic example of an ill-conditioned matrix is the Hilbert matrix of order n :

$$H_n = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ & & \vdots & & \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{pmatrix}.$$

Note that $H_n^T = H_n$ (H_n is Hermitian). Condition numbers for some Hilbert matrices appear in Table 3.1.

TABLE 3.1: Condition numbers of some Hilbert matrices

n	3	5	6	8	16	32	64
$\kappa_2(H_n)$	5×10^2	5×10^5	15×10^6	15×10^9	2.0×10^{22}	4.8×10^{46}	3.5×10^{95}

REMARK 3.33 Consider $Ax = b$. Ill-conditioning combined with round-off errors can have a disastrous effect in Gaussian elimination. Sometimes, the conditioning can be improved (κ decreased) by scaling the equations. A common scaling strategy is to *row equilibrate* the matrix A by choosing a diagonal matrix D , such that premultiplying A by D causes $\max_{1 \leq j \leq n} |a_{ij}| = 1$ for $i = 1, 2, \dots, n$. Thus, $DAx = Db$ becomes the scaled system with maximum elements in each row of DA equal to unity. (This procedure is generally recommended before Gaussian elimination with partial pivoting is employed [40]. However, there is no guarantee that equilibration with partial pivoting will not suffer greatly from effects of roundoff error.) $\square$

3.3.5.2 Roundoff Error in Gaussian Elimination

Consider the solution of $Ax = b$. On a computer, elements of A and b are represented by floating point numbers. Solving this linear system on a computer only produces an approximate solution $\hat{x}$.

There are two kinds of rounding error analysis. In *backward error analysis*, one shows that the computed solution $\hat{x}$ is the exact solution of a perturbed system of the form $(A + F)\hat{x} = b$. (See, for example, [68] or [100].) Then we have

$$Ax - A\hat{x} = -F\hat{x},$$

that is,

$$x - \hat{x} = -A^{-1}F\hat{x},$$

from which we obtain

$$\frac{\|x - \hat{x}\|_{\infty}}{\|\hat{x}\|_{\infty}} \leq \|A^{-1}\|_{\infty} \|F\|_{\infty} = \kappa_{\infty}(A) \frac{\|F\|_{\infty}}{\|A\|_{\infty}}. \quad (3.22)$$

Thus, assuming that we have estimates for $\kappa_{\infty}(A)$ and $\|F\|_{\infty}$, we can use (3.22) to estimate the error $\|x - \hat{x}\|_{\infty}$.

In *forward error analysis*, one keeps track of roundoff error at each step of the elimination procedure. Then, $x - \hat{x}$ is estimated in some norm in terms of, for example, A , $\kappa(A)$, and $\theta = \frac{p}{2}\beta^{1-t}$ [88, 89].

The analyses are lengthy and are not given here. The results, however, are useful to understand. Basically, it is shown that

$$\frac{\|F\|_{\infty}}{\|A\|_{\infty}} \leq c_n g \theta, \quad (3.23)$$

where

c_n is a constant that depends on size of the $n \times n$ matrix A ,

g is a growth factor, $g = \frac{\max_{i,j,k} |a_{ij}^{(k)}|}{\max_{i,j} |a_{ij}|}$, and

θ is the unit roundoff error, $\theta = \frac{p}{2}\beta^{1-t}$.

Note: Using backward error analysis, $c_n = 1.01n^3 + 5(n+1)^2$, and using forward error analysis, $c_n = \frac{1}{6}(n^3 + 15n^2 + 2n - 12)$.

Note: The growth factor g depends on the pivoting strategy: $g \leq 2^{n-1}$ for partial pivoting,¹⁰ while $g \leq n^{1/2}(2 \cdot 3^{1/2} \cdot 4^{1/3} \cdots n^{1/n-1})^{1/2}$ for full pivoting. (Wilkinson conjectured that this can be improved to $g \leq n$.) For example,

¹⁰It cannot be improved, since $g = 2^{n-1}$ for certain matrices.

for $n = 100$, $g \leq 2^{99} \approx 10^{30}$ for partial pivoting and $g \leq 3300$ for full pivoting.

Note: Thus, by (3.22) and (3.23), the relative error $\|x - \hat{x}\|_\infty / \|\hat{x}\|_\infty$ depends directly on $\kappa_\infty(A)$, θ , n^3 , and the pivoting strategy.

REMARK 3.34 The factor of 2^{n-1} discouraged numerical analysts in the 1950's from using Gaussian elimination, and spurred study of iterative methods for solving linear systems. However, it was found that, for most matrices, the growth factor is much less, and Gaussian elimination with partial pivoting is usually practical. $\square$

3.3.6 Iterative Refinement

We now consider a technique called *iterative refinement*, which is sometimes used to decrease the rounding error in Gaussian elimination. The following assumption is made in iterative refinement: The solution to $Ax = b$ is computed using Gaussian elimination with t digits of precision, while the solution to

$$r = b - Ax$$

is computed using $2t$ digits of precision. (r is called the *residual vector*), or, simply, *residual*.

ALGORITHM 3.4

(Iterative refinement procedure)

INPUT: A matrix $A \in \mathbb{R}^{n \times n}$, a vector $b \in \mathbb{R}^n$, and a tolerance ϵ .

OUTPUT: A refined approximation x to $Ax = b$.

1. Compute an initial approximation $x^{(0)}$ to $Ax = b$ using Gaussian elimination with t -digit arithmetic. (All multiplier and interchange information is saved to speed up later Gaussian elimination calculations.)
2. $k \leftarrow 0$.
3. DO WHILE $\|x^{(k+1)} - x^{(k)}\| > \epsilon$.
 - (a) $r^{(k)} \leftarrow b - Ax^{(k)}$. (Computed using $2t$ -digit arithmetic.)
 - (b) Solve $Ay^{(k)} = r^{(k)}$ for $y^{(k)}$, using t -digit arithmetic and using multiplier and interchange information from step 1.
 - (c) $x^{(k+1)} \leftarrow y^{(k)} + x^{(k)}$.
 - (d) $k \leftarrow k + 1$.

END DO

4. RETURN $x^{(k)}$ as x .

END ALGORITHM 3.4.

We now have the following question: using this procedure, does $x^{(k)} \rightarrow x$ as $k \rightarrow \infty$? In the analysis, it is assumed that $r^{(k)}$ is computed exactly as $2t$ digits are used.

THEOREM 3.11

Suppose that A is nonsingular, $y^{(k)}$ is the exact solution to

$$(A + F_k)y^{(k)} = r^{(k)},$$

and $r^{(k)} = b - Ax^{(k)}$ is computed exactly. If there is a constant $\gamma > 0$ for which

$$\|F_k\| \|A^{-1}\| \leq \gamma < 1/2$$

for $k = 0, 1, 2, \dots$, then the $A + F_k$, $k = 0, 1, 2, \dots$, are nonsingular, and $x_k \rightarrow x$ as $k \rightarrow \infty$.

PROOF Consider $A + F_k = A(I + A^{-1}F_k)$. We have

$$\|A^{-1}F_k\| \leq \|A^{-1}\| \|F_k\| < 1/2.$$

Thus, $I + A^{-1}F_k$ is nonsingular,

$$(I + A^{-1}F_k)^{-1} = \sum_{j=0}^{\infty} (-A^{-1}F_k)^j,$$

and

$$\|(I + A^{-1}F_k)^{-1}\| \leq \frac{1}{1 - \|A^{-1}F_k\|} \leq \frac{1}{1 - \|A^{-1}\| \|F_k\|}.$$

Hence, $A + F_k$ is nonsingular. Now consider

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + y^{(k)} = x^{(k)} + (A + F_k)^{-1}r^{(k)} \\ &= x^{(k)} + (A + F_k)^{-1}(b - Ax^{(k)}) \\ &= (A + F_k)^{-1} \left[(A + F_k)x^{(k)} + b - Ax^{(k)} \right] \\ &= (A + F_k)^{-1} \left[F_k x^{(k)} + b \right] \end{aligned}$$

Hence,

$$\begin{aligned} x^{(k+1)} - x &= (A + F_k)^{-1} \left[F_k x^{(k)} - (A + F_k)x + b \right] \\ &= (A + F_k)^{-1} F_k (x^{(k)} - x) \end{aligned}$$

Thus, $\|x^{(k+1)} - x\| \leq \|(A + F_k)^{-1} F_k\| \|x^{(k)} - x\|$. However,

$$\begin{aligned} \|(A + F_k)^{-1} F_k\| &\leq \|(A + F_k)^{-1}\| \|F_k\| = \|A^{-1}(I + A^{-1}F_k)^{-1}\| \|F_k\| \\ &\leq \frac{\|A^{-1}\| \|F_k\|}{1 - \|A^{-1}\| \|F_k\|} \\ &\leq \frac{\gamma}{1 - \gamma} < 1, \text{ since } \gamma < \frac{1}{2}. \end{aligned}$$

Thus,

$$\|x^{(k+1)} - x\| \leq \left(\frac{\gamma}{1 - \gamma}\right) \|x^{(k)} - x\| \leq \left(\frac{\gamma}{1 - \gamma}\right)^k \|x^{(-1)} - x\| \rightarrow 0 \text{ as } k \rightarrow \infty.$$

Hence, $x^{(k)} \rightarrow x$ as $k \rightarrow \infty$. $\square$

Note: In the roundoff error analysis for Gaussian elimination,

$$\|F_k\|_\infty \leq [1.01n^3 + 5(n+1)^2] \|A\|_\infty g \theta.$$

Therefore, by Theorem 3.11, it is sufficient that $\|F_k\|_\infty < 1/(2\|A^{-1}\|_\infty)$, giving the condition

$$[1.01n^3 + 5(n+1)^2] \kappa_\infty(A) g \theta < 1/2.$$

This inequality implies that iterative refinement will converge if $\theta = \frac{p}{2}\beta^{1-t}$ is sufficiently small (t large enough) and n , the condition number, and the growth factor are not too large.

Example 3.9

$$\begin{pmatrix} 3.3330 & 15920. & -10.333 \\ 2.2220 & 16.710 & 9.612 \\ 1.5611 & 5.1791 & 1.6852 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 15913. \\ 28.544 \\ 8.4254 \end{pmatrix}$$

For this problem, $\kappa_\infty(A) \approx 16000$ and the exact solution is $x = [1, 1, 1]^T$. Using Gaussian elimination with $t = 5$ and $\beta = 10$ (5-digit decimal arithmetic),

$$x_0 = [1.2001, 99991, 0.92538]^T.$$

In 10-digit arithmetic, one obtains

$$r_0 = [-0.00518, .27413, -0.18616]^T.$$

In 5-digit arithmetic, solving $Ay_0 = r_0$ gives

$$y_0 = [-0.2008, 8.9987 \times 10^{-5}, 0.074607]^T.$$

Then,

$$x_1 = x_0 + y_0 = [1.0000, 1.0000, 0.99999]^T$$

and

$$x_2 = x_1 + y_1 = [1.0000, 1.0000, 1.0000]^T.$$

□

3.3.7 Interval Bounds

In many instances, it is practical to obtain rigorous bounds on the solution x to a linear system $Ax = b$. The algorithm is a modification of the general Gaussian elimination algorithm (Algorithm 3.1) and back substitution (Algorithm 3.2), as follows.

ALGORITHM 3.5

(Interval bounds for the solution to a linear system)

INPUT: $A \in L(\mathbb{R}^n)$ and $b \in \mathbb{R}^n$.

OUTPUT: an interval vector $\mathbf{x}$ such that the exact solution to $Ax = b$ must be within the bounds $\mathbf{x}$.

1. Use Algorithm 3.1 and Algorithm 3.2 (that is, Gaussian elimination with back substitution, or any other technique) and floating point arithmetic to compute an approximation Y to A^{-1} .

2. Use interval arithmetic, with directed rounding, to compute interval enclosures to YA and Yb . That is,

(a) $\tilde{\mathbf{A}} \leftarrow YA$ (computed with interval arithmetic),

(b) $\tilde{\mathbf{b}} \leftarrow Yb$ (computed with interval arithmetic).

3. FOR $k = 1, 2, \dots, n-1$ (forward phase using interval arithmetic)

FOR $i = k+1, \dots, n$

(a) $\mathbf{m}_{ik} \leftarrow \tilde{\mathbf{a}}_{ik} / \tilde{\mathbf{a}}_{kk}$.

(b) $\tilde{\mathbf{a}}_{ik} \leftarrow [0, 0]$.

(c) FOR $j = k+1, \dots, n$

$\tilde{\mathbf{a}}_{ij} \leftarrow \tilde{\mathbf{a}}_{ij} - \mathbf{m}_{ik} \tilde{\mathbf{a}}_{kj}$.

END FOR

(d) $\tilde{\mathbf{b}}_i \leftarrow \tilde{\mathbf{b}}_i - \mathbf{m}_{ik} \tilde{\mathbf{b}}_k$.

END FOR

END FOR

$$4. \mathbf{x}_n \leftarrow \tilde{\mathbf{b}}_n / \tilde{\mathbf{a}}_{nn}.$$

5. FOR $k = n - 1, n - 2, \dots, 1$ (*back substitution*)

$$\mathbf{x}_k \leftarrow (\tilde{\mathbf{b}}_k - \sum_{j=k+1}^n \tilde{\mathbf{a}}_{kj} \mathbf{x}_j) / \tilde{\mathbf{a}}_{kk}.$$

END FOR

END ALGORITHM 3.5.

Note: We can explicitly set $\mathbf{a}_{ik}$ to zero without loss of mathematical rigor, even though, using interval arithmetic, $\mathbf{a}_{ik} - \mathbf{m}_{ik} \mathbf{a}_{kk}$ may not be exactly $[0, 0]$. In fact, this operation does not even need to be done, since we need not reference $\mathbf{a}_{ik}$ in the back substitution process.

Note: Obtaining the rigorous bounds $\mathbf{x}$ in Algorithm 3.2 is more costly than computing an approximate solution with floating point arithmetic using Gaussian elimination with back substitution, because an approximate inverse Y must explicitly be computed to precondition the system. However, both computations take $\mathcal{O}(n^3)$ operations for general systems.

The following theorem clarifies why we may use Algorithm 3.5 to obtain mathematically rigorous bounds.

THEOREM 3.12

Define the solution set to $\tilde{\mathbf{A}}\mathbf{x} = \tilde{\mathbf{b}}$ to be

$$\Sigma(\tilde{\mathbf{A}}, \tilde{\mathbf{b}}) = \left\{ \mathbf{x} \mid \tilde{\mathbf{A}}\mathbf{x} = \tilde{\mathbf{b}} \text{ for some } \tilde{\mathbf{A}} \in \tilde{\mathbf{A}} \text{ and } \tilde{\mathbf{b}} \in \tilde{\mathbf{b}} \right\}.$$

If $A\mathbf{x}^* = \mathbf{b}$, then $\mathbf{x}^* \in \Sigma(\tilde{\mathbf{A}}, \tilde{\mathbf{b}})$. Furthermore, if $\mathbf{x}$ is the output to Algorithm 3.5, then $\Sigma(\tilde{\mathbf{A}}, \tilde{\mathbf{b}}) \subseteq \mathbf{x}$.

For facts enabling a proof of Theorem 3.12, see [62] or other references on interval analysis.

Example 3.10

Let the matrix A and the right-hand-side vector $\mathbf{b}$ be as in Example 3.9 (on page 129). We will use IEEE double precision (see Table 1.1 on page 19) to compute Y , and we will use interval arithmetic based on IEEE double

precision.¹¹ Rounded to 14 decimal digits,¹² we obtain

$$Y \approx \begin{pmatrix} -0.00012055643706 & -0.14988499865822 & 0.85417095741675 \\ 0.00006278655296 & 0.00012125786211 & -0.00030664438576 \\ -0.00008128244868 & 0.13847464088044 & -0.19692507695527 \end{pmatrix}.$$

Using outward rounding in both the computation and the decimal display, we obtain

$$\tilde{A} \subseteq \begin{pmatrix} [1.00000000000000, 1.00000000000000] & [-0.00000000000012, -0.00000000000011] \\ [0.00000000000000, 0.00000000000001] & [1.00000000000000, 1.00000000000001] \\ [0.00000000000000, 0.00000000000001] & [0.00000000000013, 0.00000000000014] \end{pmatrix} \\ \begin{pmatrix} [-0.00000000000001, -0.00000000000000] \\ [-0.00000000000001, -0.00000000000000] \\ [0.99999999999999, 1.00000000000001] \end{pmatrix},$$

and

$$\tilde{b} \subseteq \begin{pmatrix} [0.99999999999988, 0.99999999999989] \\ [1.00000000000000, 1.00000000000001] \\ [1.00000000000013, 1.00000000000014] \end{pmatrix}.$$

Completing the remainder of Algorithm 3.5 then gives

$$x^* \in x \subseteq \begin{pmatrix} [0.99999999999999, 1.00000000000001] \\ [0.99999999999999, 1.00000000000001] \\ [0.99999999999999, 1.00000000000001] \end{pmatrix}.$$

□

Note: There are various alternate ways of using interval arithmetic to obtain rigorous bounds on the solution set to linear systems of equations. Some of these are related mathematically to the interval Newton method introduced in §2.5 on page 54, while others are related to the iterative techniques we discuss later in this section. The effectiveness and practicality of a particular such technique depend on the condition of the system, and whether the entries in the matrix A and right hand side vector b are points to start, or whether there are larger uncertainties in them (that is, whether or not these coefficients are wide or narrow intervals). A good theoretical reference is [62] and some additional practical detail is given in our monograph [44].

We now consider another direct method for computing the solution of a linear system $Ax = b$.

3.3.8 Orthogonal Decomposition (QR Decomposition)

This direct method for computing the solution of $Ax = b$ is based on *orthogonal decomposition*, also known as the *QR decomposition* or *QR factorization*.

¹¹We used MATLAB for our actual computations, and we used the INTLAB toolbox for MATLAB for the interval arithmetic.

¹²as MATLAB displays it

In addition to solving linear systems, the QR factorization is also useful in least squares problems and eigenvalue computations. The goal of orthogonal decomposition is to find $A = QR$, where Q is orthogonal (i.e., $Q^T Q = I$) and R is upper triangular. Then $Ax = b$ has the form $QRx = b$, so $Rx = Q^T b$, and this can be rapidly solved using backsolving.

We will assume in this section that A is a real matrix.

There are several common ways of computing QR decompositions.

3.3.8.1 QR Decomposition with Householder Transformations

We will first show how to compute the QR decomposition with Householder transformations.

DEFINITION 3.30 Let $u \in \mathbb{R}^n$ and suppose that $\|u\|_2^2 = 1 = u^T u$. Then the $n \times n$ matrix $U = I - 2uu^T$ is called a Householder transformation.

As proved in the following lemmas, Householder transformations have several interesting properties.

LEMMA 3.1

Let U be a Householder transformation. Then $U^T = U$, $U^T U = I$ (U is orthogonal) and $U^2 = I$ (U is involutory).

PROOF Clearly $U^T = U$. Then

$$U^T U = U^2 = (I - 2uu^T)^2 = I - 4uu^T + 4uu^T uu^T = I,$$

since $u^T u = 1$. □

LEMMA 3.2

Let $u \in \mathbb{R}^n$, $u \neq 0$, and $\theta = \frac{1}{2}\|u\|_2^2$. Then

$$U = I - \frac{uu^T}{\theta}$$

is a Householder transformation.

PROOF Let $v = (1/\|u\|_2)u$, so $\|v\|_2 = 1$. Then,

$$U = I - (1/\theta)uu^T = I - 2vv^T,$$

so U is a Householder transformation. □

The next lemma shows that Householder transformations have the important capability of introducing zeros into vectors.

LEMMA 3.3

Let $x \in \mathbb{R}^n$, $x \neq 0$, and let x_1 be the first element of x . Let

$$\sigma = \operatorname{sgn}(x_1) \|x\|_2$$

(where $\operatorname{sgn}(0) = 1$), let

$$u = x + \sigma e_1 \quad \text{and} \quad \theta = \frac{1}{2} \|u\|_2^2.$$

Then

$$U = I - (1/\theta)uu^T$$

is a Householder transformation, and $Ux = -\sigma e_1$.

PROOF Note that $u = x + \sigma e_1 \neq 0$, since $x \neq -\sigma e_1$. Thus, by Lemma 3.2, U is a Householder transformation. Consider $x^T x = \sigma^2$ and

$$\begin{aligned} \theta &= \frac{1}{2} \|u\|_2^2 = \frac{1}{2} u^T u \\ &= \frac{1}{2} (x + \sigma e_1)^T (x + \sigma e_1) = \frac{1}{2} (x^T x + \sigma x^T e_1 + \sigma^2 e_1^T e_1 + \sigma e_1^T x) \\ &= \frac{1}{2} (2\sigma^2 + 2\sigma x_1) = \sigma^2 + \sigma x_1. \end{aligned}$$

Then,

$$\begin{aligned} Ux &= x - \frac{1}{\theta} uu^T x = x - \left[\frac{(x + \sigma e_1)(x + \sigma e_1)^T x}{\sigma^2 + \sigma x_1} \right] \\ &= x - (x + \sigma e_1) \left[\frac{\sigma^2 + \sigma x_1}{\sigma^2 + \sigma x_1} \right] = -\sigma e_1. \end{aligned}$$

□

To avoid problems with underflows and overflows in computation of U , x is scaled at the beginning of the computation. The algorithm proceeds as follows.

ALGORITHM 3.6

(Computing a Householder transformation.)

INPUT: $x \in \mathbb{R}^n$ with $x \neq 0$.

OUTPUT: σ , θ , and u such that $Ux = (I - uu^T/\theta)x = -\sigma e_1$.

1. $v = x/\|x\|_\infty$ (i.e., $\|v\|_\infty = 1$).
2. $\tilde{\sigma} = \operatorname{sgn}(v_1) \|v\|_2$.
3. $u_1 = v_1 + \tilde{\sigma}$, $u_i = v_i$ for $2 \leq i \leq n$.

$$4. \theta = \tilde{\sigma}u_1. \quad (\theta = \tilde{\sigma}^2 + \tilde{\sigma}v_1).$$

$$5. \sigma = \tilde{\sigma}\|x\|_\infty \quad (\text{Note that } \sigma = \text{sgn}(x_1)\|x\|_2).$$

END ALGORITHM 3.6.

Steps 1 and 5 of Algorithm 3.6 are to avoid catastrophic underflows and overflows in the computation of $\|v\|_2$ in Step 2. See Example 1.16 on page 21.

Householder transformations can also be defined for vectors $u \in \mathbb{C}^n$:

DEFINITION 3.31 *Let $u \in \mathbb{C}^n$ be such that $\|u\|_2^2 = u^H u = 1$ and define $U = I - 2uu^H$. Then $U^H = U$ and $U^H U = I$. (Such a complex Householder transformation U is called a complex reflector.)*

The following useful result for complex reflectors is analogous to Lemma 3.3, that is, it allows us to transform a complex vector to a unit vector.

LEMMA 3.4

Let $x \in \mathbb{C}^n$, $x \neq 0$, $x_1 = re^{i\tilde{\theta}}$, and $\sigma = e^{i\tilde{\theta}}\|x\|_2$ (with $\tilde{\theta} = 0$ if $x_1 = 0$). Let $u = x + \sigma e_1$, $\theta = \frac{1}{2}\|u\|_2^2$, and $U = I - uu^H/\theta$. Then $Ux = -\sigma e^{(1)}$.

PROOF Note that $x^H x = \overline{\sigma}\sigma$ and

$$\begin{aligned} \theta &= \frac{1}{2}u^H u = \frac{1}{2} \left[x^H + \overline{\sigma}e^{(1)H} \right] \left[x + \sigma e^{(1)} \right] \\ &= \frac{1}{2} \left[x^H x + \overline{\sigma}x_1 + \sigma\overline{x}_1 + \overline{\sigma}\sigma \right] \\ &= \|x\|_2^2 + r\|x\|_2. \end{aligned}$$

Now,

$$\begin{aligned} Ux &= x - \frac{u}{\theta}[u^H x] \\ &= x - \frac{u}{\theta}[x^H x + \overline{\sigma}x_1] \\ &= x - \frac{u}{[\|x\|_2^2 + r\|x\|_2]}[\|x\|_2^2 + r\|x\|_2] \\ &= x - u \\ &= -\sigma e_1. \end{aligned}$$

□

Now consider the transformation of a matrix A into upper trapezoidal form using Householder transformations. It is assumed here that A is $m \times n$, with $m \geq n$. We will find a sequence of Householder transformations that will

transform A into upper trapezoidal form, i.e., into the form $\begin{pmatrix} R \\ 0 \end{pmatrix}$, where R is an $n \times n$ upper triangular matrix and 0 is an $(m - n) \times n$ matrix of zeros.

THEOREM 3.13

Let A be a real $m \times n$ matrix with $m \geq n$. Then there exist Householder transformations $U_1, U_2, \dots, U_r$ such that $A_{r+1} = U_r U_{r-1} \cdots U_1 A$ is upper trapezoidal, where $r = \min(m - 1, n)$.

PROOF At the k -th stage, we construct a matrix U_k such that multiplication on the left by U_k introduces zeros below the diagonal in the k -th column. Let $A = A^{(1)} = [c^{(1)}, M_1]$, where $c^{(1)} \in \mathbb{R}^m$ and M_1 is an $m \times (n - 1)$ matrix. Using Algorithm 3.6, we construct a Householder transformation $U_1 = I - u^{(1)}u^{(1)T}/\theta_1$ such that $U_1 c^{(1)} = r_{11}(1, 0, \dots, 0)^T \in \mathbb{R}^m$. Then

$$A^{(2)} = U_1 A^{(1)} = \left(r_{11} e^{(1)} \left| U_1 M_1 \right. \right) = \left(\begin{array}{c|c} \begin{matrix} r_{11} \\ 0 \\ \vdots \\ 0 \end{matrix} & U_1 M_1 \end{array} \right).$$

We now suppose that at the k -th step we have

$$A^{(k)} = U_{k-1} U_{k-2} \cdots U_1 A^{(1)} = \left(\begin{array}{c|c|c} R_k & r^{(k)} & B_k \\ \hline 0 & c^{(k)} & M_k \end{array} \right),$$

where R_k is $(k - 1) \times (k - 1)$ upper triangular, $r^{(k)} \in \mathbb{R}^{k-1}$, $c^{(k)} \in \mathbb{R}^{m-k+1}$, B_k is $(k - 1) \times (n - k)$, M_k is $(m - k + 1) \times (n - k)$, and “0” represents an $(n - k + 1) \times (k - 1)$ matrix consisting entirely of zeros. Using Algorithm 3.6, we find $u^{(k)} \in \mathbb{R}^{m-k+1}$ and θ_k such that the $(m - k + 1) \times (m - k + 1)$ matrices $U'_k = I - u^{(k)}u^{(k)T}/\theta_k$ are Householder transformations and $U'_k c^{(k)} = r_{kk}(1, 0, \dots, 0)^T \in \mathbb{R}^{m-k+1}$. We define

$$U_k = \left(\begin{array}{c|c} I_{k-1} & 0 \\ \hline 0 & U'_k \end{array} \right).$$

U_k is a Householder transformation, since, if $U'_k = I_{m-k+1} - 2vv^T$ where $\|v\|_2 = 1$, then $U_k = I_m - 2ww^T$, where $w = (0, \dots, 0, v^T)^T \in \mathbb{R}^m$. Now, using the above matrices,

$$\begin{aligned} A^{(k+1)} &= U_k A^{(k)} = \left(\begin{array}{c|c|c} R_k & r^{(k)} & B_k \\ \hline 0 & U'_k c^{(k)} & U'_k M_k \end{array} \right) \\ &= \left(\begin{array}{c|c|c} R_k & r^{(k)} & B_k \\ \hline 0 & r_{kk} & U'_k M_k \\ \vdots & \vdots & \\ 0 & 0 & \end{array} \right) = \left(\begin{array}{c|c|c} R_{k+1} & r^{(k+1)} & B_{k+1} \\ \hline 0 & c^{(k+1)} & M_{k+1} \end{array} \right). \end{aligned}$$

where R_{k+1} is upper triangular, i.e.,

$$R_{k+1} = \left(\begin{array}{c|c} R_k & r^{(k)} \\ \hline 0 & r_{kk} \end{array} \right).$$

In general, we need $r - 1$ steps to reduce A to upper triangular form, where $r \leq \min\{m, n\}$ is the rank of A . $\square$

Note: To compute the QR factorization of an $m \times n$ matrix A , where $m > n$ and the rank of A is $r = n$, we denote the product of the Householder transformations by Q , that is, $Q = U_r U_{r-1} \dots U_1$. We then partition the upper trapezoidal matrix $A^{(r+1)}$ into either

$$QA = A^{(r+1)} = \left(\begin{array}{c} R \\ 0 \end{array} \right),$$

where R is an $n \times n$ upper triangular matrix. Observe that Q is orthogonal, since

$$Q^T Q = U_1^T U_2^T \dots U_r^T U_r \dots U_1 = I.$$

We now partition the $m \times m$ matrix Q^T into the form $Q^T = (Q_1 | Q_2)$, where Q_1 is $m \times n$ and Q_2 is $m \times (m - n)$. Notice that Q_1 has orthonormal columns because Q has orthonormal columns. Since

$$A = Q^T A^{(r+1)} = (Q_1 | Q_2) \left(\begin{array}{c} R \\ 0 \end{array} \right) = Q_1 R,$$

$A = Q_1 R$ is the desired QR factorization of matrix A . Notice that R is an $n \times n$ upper triangular matrix and Q_1 is an $m \times n$ matrix with orthonormal columns.

Note: In the special case where $m = n$, then $A = QR$ where Q and R are $n \times n$. Then, $Ax = b$ can be solved by backsolving $Rx = Q^T b$. However, this algorithm, which produces the QR factorization, requires $\mathcal{O}(\frac{2}{3}n^3)$ multiplications and divisions. This method is, however, generally more stable with respect to rounding errors [40].

3.3.8.2 QR Decomposition with Givens Rotations

The QR factorization can also be computed using Givens transformations, which can introduce zeros at any position in a matrix. A sequence of Givens transformations can thus be used to transform A to upper triangular form, i.e., $P_s P_{s-1} \dots P_1 A = R$, where $Q = \Pi_{i=1}^s P_i$. To understand Givens transformations (plane rotations), consider first $x \in \mathbb{R}^2$, and let

$$\gamma = \frac{x_1}{\|x\|_2} \quad \text{and} \quad \sigma = \frac{x_2}{\|x\|_2}.$$

Then

$$\begin{pmatrix} \gamma & \sigma \\ -\sigma & \gamma \end{pmatrix} x = \begin{pmatrix} \|x\|_2 \\ 0 \end{pmatrix}.$$

To introduce a zero into the j -th location of a vector x of length n , let

$$P_{ij} = \left(\begin{array}{c|c|c|c|c} 1 & & i\text{-th} & & j\text{-th} \\ & \ddots & \text{col.} & & \text{col.} \\ & & \downarrow & & \downarrow \\ & & 1 & & \\ \hline i\text{-th} & \rightarrow & \gamma & & \sigma \\ \text{row} & & & & \\ \hline & & & 1 & \\ & & & \ddots & \\ & & & & 1 \\ \hline j\text{-th} & \rightarrow & -\sigma & & \gamma \\ \text{row} & & & & \\ \hline & & & & 1 \\ & & & & \ddots \\ & & & & & 1 \end{array} \right),$$

where

$$\gamma = \frac{x_i}{\sqrt{x_i^2 + x_j^2}} \text{ and } \sigma = \frac{x_j}{\sqrt{x_i^2 + x_j^2}}.$$

Then it is straightforward to show that $P_{ij}P_{ij}^T = I$ (P_{ij} is orthogonal), and

$$P_{ij}x = \left(x_1, x_2, \dots, x_{i-1}, \sqrt{x_i^2 + x_j^2}, x_{i+1}, \dots, x_{j-1}, 0, x_{j+1}, \dots, x_n \right)^T.$$

3.3.8.3 QR Decomposition with the Gram–Schmidt Procedure

A third way to obtain the QR factorization is with a Gram–Schmidt orthogonalization procedure. Let $A = (a_1, a_2, \dots, a_n)$, where $a_i \in \mathbb{R}^m$ for $i = 1, 2, \dots, n$ and $m \geq n$, and assume that $a_1, \dots, a_n$ are linearly independent. In the Gram–Schmidt process, let

$$\begin{cases} q_1 = a_1 \\ q_k = a_k - \sum_{j=1}^{k-1} \alpha_{jk} q_j, \quad \alpha_{jk} = \frac{(q_j, a_k)}{(q_j, q_j)}, \quad k = 2, 3, \dots, n. \end{cases}$$

Then the q_i , $i = 1, 2, \dots, n$ are orthogonal. To see why, consider

$$q_2 = a_2 - \alpha_{12} q_1, \quad \alpha_{12} = \frac{(q_1, a_2)}{(q_1, q_1)}.$$

Then $(q_1, q_2) = (q_1, a_2) - \alpha_{12}(q_1, q_1) = 0$. Also, $q_2 \neq 0$ because a_2 is independent from a_1 . Continuing in this way, it can be verified by induction the entire set of q_k is orthogonal.

Now notice that

$$\begin{aligned} q_1 &= a_1 \\ q_2 &= a_2 - \alpha_{12}a_1 \\ q_3 &= a_3 - \alpha_{13}a_1 - \alpha_{23}a_2 \\ &\vdots \end{aligned}$$

so we can write

$$q_i = \sum_{j=1}^i t_{ji}a_j \text{ for } i = 1, 2, \dots, n. \quad (3.24)$$

Now consider $A = QR = [q_1, q_2, \dots, q_n]R$, so $AR^{-1} = Q$. However, writing (3.24) in matrix form, we obtain

$$(a_1, a_2, \dots, a_n) \begin{pmatrix} t_{11} & t_{12} & \cdots & t_{1n} \\ 0 & t_{22} & & \vdots \\ \vdots & & \ddots & \\ 0 & \cdots & 0 & t_{nn} \end{pmatrix} = (q_1, q_2, \dots, q_n),$$

which gives $t_{11}a_1 = q_1$, $t_{12}a_1 + t_{22}a_2 = q_2, \dots$. This shows that R^{-1} can be chosen to have entries that are the coefficients α_{jk} computed in (3.24), and thus that the Gram–Schmidt process is equivalent to finding a QR decomposition of A . However, we see that

$$Q^T Q = \text{diag}(q_1^T q_1, q_2^T q_2, \dots, q_n^T q_n) = \Lambda \neq I.$$

Nonetheless, setting $\hat{Q} = Q\Lambda^{-\frac{1}{2}}$ and $\hat{R} = \Lambda^{\frac{1}{2}}R$, we obtain

$$\hat{Q}\hat{R} = Q\Lambda^{-\frac{1}{2}}\Lambda^{\frac{1}{2}}R = A, \text{ where } \hat{Q}^T\hat{Q} = \Lambda^{-\frac{1}{2}}Q^TQ\Lambda^{-\frac{1}{2}} = I.$$

We treat the Gram–Schmidt process somewhat more formally, in the context of general inner product spaces, in Section 4.2.3 on page 201.

REMARK 3.35 The Gram–Schmidt process as we have just explained it is numerically unstable. However, it can be modified to take account of the effects of roundoff error. This *modified Gram–Schmidt* process is sometimes used in practice [78]. $\square$

3.3.8.4 Least Squares and the QR Decomposition

Overdetermined linear systems (with more equations than unknowns) occur frequently in data fitting, in mathematical modeling and statistics. For example, we may have data of the form $\{(t_i, y_i)\}_{i=1}^m$, and we wish to model

the dependence of y on t by a linear combination of n basis functions $\{\varphi_j\}_{j=1}^n$, that is,

$$y \approx f(t) = \sum_{i=1}^n x_i \varphi_i(t), \quad (3.25)$$

where $m > n$. Setting $f(t_i) = y_i$, $1 \leq i \leq m$, gives the overdetermined linear system

$$\begin{pmatrix} \varphi_1(t_1) & \varphi_2(t_1) & \cdots & \varphi_n(t_1) \\ \varphi_1(t_2) & \varphi_2(t_2) & \cdots & \varphi_n(t_2) \\ & & \ddots & \\ \varphi_1(t_m) & \varphi_2(t_m) & \cdots & \varphi_n(t_m) \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}, \quad (3.26)$$

that is,

$$Ax = b, \text{ where } A \in L(\mathbb{R}^n, \mathbb{R}^m), a_{ij} = \varphi_j(t_i), \text{ and } b_i = y_i. \quad (3.27)$$

Perhaps the most common way of fitting data is with *least squares*, in which we find x^* such that

$$\frac{1}{2} \|Ax^* - b\|_2^2 = \min_{x \in \mathbb{R}^n} \varphi(x), \quad \text{where } \varphi(x) = \frac{1}{2} \|Ax - b\|_2^2. \quad (3.28)$$

(Note that x^* minimizes the 2-norm of the residual vector $r(x) = Ax - b$, since the function $g(u) = u^2$ is increasing.)

The naive way of finding x^* is to set the gradient $\nabla \varphi(x) = 0$ and simplify. Doing so gives the *normal equations*:

$$A^T Ax = A^T b. \quad (3.29)$$

(See Exercise 28 on page 183.) However, the normal equations tend to be very ill-conditioned. For example, if $m = n$, $\kappa_2(A^T A) = \kappa_2(A)^2$. Fortunately, the least squares solution x^* may be computed with a *QR* decomposition. In particular,

$$\|Ax - b\|_2 = \|QRx - b\|_2 = \|Q^T(QRx - b)\|_2 = \|Rx - Q^T b\|_2.$$

(Above, we used $\|Ux\|_2 = \|x\|_2$ when U is unitary; see Exercise 25 on page 182 below.) However,

$$\|Rx - Q^T b\|_2^2 = \sum_{i=1}^n \left(\left\{ \sum_{j=1}^i r_{ij} x_j \right\} - (Q^T b)_i \right)^2 + \sum_{i=m+1}^n (Q^T b)_i^2. \quad (3.30)$$

Observe now:

1. All m terms in the sum in (3.30) are nonnegative.

2. The first n terms can be made exactly zero.
3. The last $m - n$ terms are constant.

Therefore,

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2 = \sum_{i=m+1}^n (Q^T b)_i^2,$$

and the minimizer x^* can be computed by backsolving the square triangular system consisting of the first n rows of $Rx = Q^T b$.

We will say more about least squares approximations in Chapter 4, in Section 4.3.6. We now turn to iterative techniques for linear systems of equations.

3.4 Iterative Methods for Solving Linear Systems

Here, we study iterative solution of linear systems

$$Ax = b, \quad \text{i.e.} \quad \sum_{k=1}^n a_{jk} x_k = b_j, \quad j = 1, 2, \dots, n. \quad (3.31)$$

Good references for iterative solution of linear systems are [49, 68, 96, 103].

Why may we wish to solve (3.31) iteratively? Suppose that $n = 10,000$ or more, which is not unreasonable for many problems. Then A has 10^8 elements, making it difficult to store or solve (3.31) directly using, for example, Gaussian elimination. A simple way to obtain iterative methods is to split A as

$$A = M - N, \quad (3.32)$$

with M nonsingular. Then (3.31) becomes

$$Mx = Nx + b. \quad (3.33)$$

Now suppose that it is “easy” to solve $My = q$ by a direct method, for example, if M is triangular. Then given $x^{(0)}$, we generate iterates by solving

$$Mx^{(k+1)} = Nx^{(k)} + b \quad \text{for } k = 0, 1, 2, \dots \quad (3.34)$$

If $x^{(k)} \rightarrow x$ as $k \rightarrow \infty$, then (3.34) gives $Ax = b$, that is, the limit vector is the solution to the original problem. Define the iteration matrix B by

$$B = M^{-1}N \text{ and } c = M^{-1}b. \quad (3.35)$$

Then (3.34) has the form

$$x^{(k+1)} = Bx^{(k)} + c, \quad k = 0, 1, 2, \dots, \quad (3.36)$$

where $x^{(0)}$ is an initial guess. If $x^{(k)} \rightarrow x$ as $k \rightarrow \infty$, the solution x satisfies

$$x = Bx + c,$$

so $x = M^{-1}Nx + c$, so $Mx = Nx + b$, so $Ax = b$.

Note: Iterates defined by (3.36) can be viewed as fixed point iterates that under certain conditions converge to the fixed point.

DEFINITION 3.32 *The iterative method defined by (3.36) is called convergent if, for all initial values $x^{(0)}$, we have $x^{(k)} \rightarrow A^{-1}b$ as $k \rightarrow \infty$.*

Let $\epsilon^{(k)} = x^{(k)} - x$ for $k = 0, 1, 2, \dots$ be the errors in the k -th iterate. Then, since $x = Bx + c$ and $x^{(k+1)} = Bx^{(k)} + c$, we have

$$\epsilon^{(k+1)} = B\epsilon^{(k)}, \quad \text{that is, } \epsilon^{(k+1)} = B^{k+1}\epsilon^{(0)} \quad \text{for } k = 0, 1, 2, \dots \quad (3.37)$$

We see that $x^{(k)} \rightarrow x$ is equivalent to $\epsilon^{(k)} \rightarrow 0$. Thus, the iterative method is convergent if and only if $B^k\epsilon^{(0)} \rightarrow 0$ as $k \rightarrow \infty$. We have the following result.

THEOREM 3.14

Let x be the solution of $Ax = b$. The following are equivalent:

- (a) *the iterative method (3.36) is convergent, i.e., for all $x^{(0)}$ we have $x^{(k)} \rightarrow x$ as $k \rightarrow \infty$.*
- (b) *the spectral radius obeys $\rho(B) < 1$;*
- (c) *there exists a matrix norm $\|\cdot\|$ such that $\|B\| < 1$.*

PROOF Recall first that Theorem 3.5 on page 102 states that the following are equivalent:

- (a) $\lim_{k \rightarrow \infty} A^k = 0$.
- (b) $\lim_{k \rightarrow \infty} A^k x = 0 \quad \forall x \in \mathbb{C}^n$.
- (c) $\rho(A) < 1$.

- (d) There exists a matrix norm $\|\cdot\|$ such that $\|A\| < 1$.

We will show $(a) \Rightarrow (b) \Rightarrow (c) \Rightarrow (a)$. Let the iterative method (3.36) be convergent, and let y be a given vector. Let $x^{(0)} = y + x$ where $x = A^{-1}b$. Since (3.36) is convergent, $B^k y \rightarrow 0$ as $k \rightarrow \infty$ for any y . Hence, applying Theorem 3.5 gives $(a) \Rightarrow (b) \Rightarrow (c)$. Finally, suppose that (c) is given; then $\|\epsilon^{(k+1)}\| \leq \|B\|^k \|\epsilon^{(0)}\| \rightarrow 0$, so $(c) \Rightarrow (a)$. $\square$

We study first three basic iterative methods: Jacobi, Gauss–Seidel, and SOR. Given an $n \times n$ matrix A , we can split A into $A = L + D + U$, where L is strictly lower triangular, D is diagonal, and U is strictly upper triangular. Specifically,

$$L = \begin{pmatrix} 0 & \cdots & 0 \\ a_{21} & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{pmatrix}, \quad U = \begin{pmatrix} 0 & a_{12} & \cdots & a_{1n} \\ \vdots & \ddots & \ddots & \vdots \\ & & a_{n-1,n} & \\ 0 & \cdots & & 0 \end{pmatrix}, \quad (3.38)$$

and $D = \text{diag}(a_{11}, a_{22}, \dots, a_{nn})$.

3.4.1 The Jacobi Method

We first consider the *Jacobi*, or *total step* method.” Let

$$M = D \text{ and } N = -(L + U), \quad (3.39)$$

where M and N are as in (3.34), and L , D , and U are as in (3.38). If $a_{ii} \neq 0$ for $i = 1, 2, \dots, n$, then M is nonsingular, and $B = M^{-1}N$ has the form

$$B = -D^{-1}(L + U) \equiv J. \quad (3.40)$$

J is called the iteration matrix for the “point” Jacobi method (versus the iteration matrix for the “block” Jacobi method). The iterative method becomes:

$$x^{(k+1)} = -D^{-1}(L + U)x^{(k)} + D^{-1}b, \quad k = 0, 1, 2, \dots \quad (3.41)$$

Generally, one uses the following equations to solve for $x^{(k+1)}$:

$$\begin{aligned} x_i^{(0)} & \text{ is given} \\ x_i^{(k+1)} &= \frac{1}{a_{ii}} \left\{ b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right\}, \end{aligned} \quad (3.42)$$

for $k \geq 0$ and $1 \leq i \leq n$. Equations (3.42) are easily programmed.

Note: One can think of the Jacobi method as simply solving the i -th equation of the system $Ax = b$ for the i -th variable, and substituting the old values of the other variables to get new values of the i -th variable.

3.4.2 The Gauss–Seidel Method

We now discuss the *Gauss–Seidel* method, or *successive relaxation* method. If in the Jacobi method, we use the new values of x_j as they become available,

then

$$x_i^{(0)} \quad \text{is given,}$$

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left\{ b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij} x_j^{(k)} \right\}, \quad (3.43)$$

for $k \geq 0$ and $1 \leq i \leq n$. (We continue to assume that $a_{ii} \neq 0$ for $i = 1, 2, \dots, n$.) The iterative method (3.43) is called the Gauss–Seidel method, and can be obtained in matrix form by letting $M = L + D$ and $N = -U$. Then

$$B = M^{-1}N = -(L + D)^{-1}U \equiv \mathcal{G},$$

and

$$x^{(k+1)} = -(L + D)^{-1}Ux^{(k)} + (L + D)^{-1}b \text{ for } k \geq 0. \quad (3.44)$$

The matrix $\mathcal{G}$ is called the (point) Gauss–Seidel matrix.

Note: The Gauss–Seidel method only requires storage of

$$[x_1^{(k+1)}, x_2^{(k+1)}, \dots, x_{i-1}^{(k+1)}, x_i^{(k)}, x_{i+1}^{(k)}, \dots, x_n^{(k)}]^T$$

to compute $x_i^{(k+1)}$. The Jacobi method requires storage of $x^{(k)}$ as well as $x^{(k+1)}$. Also, the Gauss–Seidel method generally converges faster. For these reasons, the Jacobi method is seldom used in practice.¹³

Example 3.11

$$\begin{pmatrix} 2 & 1 \\ -1 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \end{pmatrix}, \quad \text{that is,} \quad \begin{array}{rcl} 2x_1 + x_2 & = & 3 \\ -x_1 + 3x_2 & = & 2. \end{array}$$

(The exact solution is $x_1 = x_2 = 1$.) The Jacobi and Gauss–Seidel methods have the forms

$$\text{Jacobi: } \begin{cases} x_1^{(k+1)} = \frac{3}{2} - \frac{1}{2}x_2^{(k)} \\ x_2^{(k+1)} = \frac{2}{3} + \frac{1}{3}x_1^{(k)} \end{cases},$$

$$\text{Gauss–Seidel: } \begin{cases} x_1^{(k+1)} = \frac{3}{2} - \frac{1}{2}x_2^{(k)} \\ x_2^{(k+1)} = \frac{2}{3} + \frac{1}{3}x_1^{(k+1)} \end{cases}.$$

The results in Table 3.2 are obtained with $x^{(0)} = (0, 0)^T$. Observe that the Gauss–Seidel method converges roughly twice as fast as the Jacobi method. This behavior is provable. $\square$

¹³Methods that are a kind of hybrid between Jacobi and Gauss–Seidel are implemented in

TABLE 3.2: Iterates of the Jacobi and Gauss–Seidel methods, for Example 3.11

k	$x_1^{(k)}$ Jacobi	$x_2^{(k)}$ Jacobi	$x_1^{(k)}$ G–S	$x_2^{(k)}$ G–S
0	0	0	0	0
1	1.5	0.667	1.5	1.167
2	1.167	1.167	0.917	0.972
3	0.917	1.056	1.014	1.005
4	0.972	0.972	0.998	0.999
5	1.014	0.991	1.000	1.000
6	1.005	1.005		
7	0.998	1.002		
8	0.999	0.999		
9	1.000	1.000		

3.4.3 Successive Overrelaxation

We now describe *Successive OverRelaxation* (SOR). In the SOR method, one computes $x_i^{(k+1)}$ to be a weighted mean of $x_i^{(k)}$ and the Gauss–Seidel iterate for that element. Specifically, for $\sigma \neq 0$ a real parameter, the SOR method is given by

$x_i^{(0)}$ is given

$$x_i^{(k+1)} = (1 - \sigma)x_i^{(k)} + \frac{\sigma}{a_{ii}} \left\{ b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right\}, \quad (3.45)$$

for $1 \leq i \leq n$ and for $k \geq 0$. The parameter σ is called a relaxation factor. If $\sigma < 1$, we call σ an *underrelaxation* factor and if $\sigma > 1$, we call σ an *overrelaxation* factor. Note that if $\sigma = 1$, the Gauss–Seidel method is obtained.

Note: For certain classes of matrices and certain σ between 1 and 2, the SOR method converges faster than the Gauss–Seidel method.

We can write (3.45) in the matrix form:

$$\left(L + \frac{1}{\sigma}D \right) x^{(k+1)} = - \left\{ U + \left(1 - \frac{1}{\sigma} \right) D \right\} x^{(k)} + b \quad (3.46)$$

parallel processing environment, where separate processors are working simultaneously on separate sets of indices i , and it may not be efficient to communicate new values as soon as they are computed.

for $k = 0, 1, 2, \dots$, with $x^{(0)}$ given. Thus, letting $M = L + (1/\sigma)D$ and $N = -[U + (1 - (1/\sigma))D]$, we see that (3.46) is of the form (3.34). Thus,

$$B = M^{-1}N = (\sigma L + D)^{-1}[(1 - \sigma)D - \sigma U] \equiv \mathcal{S}_\sigma,$$

and

$$x^{(k+1)} = \mathcal{S}_\sigma x^{(k)} + \left(L + \frac{1}{\sigma}D\right)^{-1} b. \quad (3.47)$$

The matrix $\mathcal{S}_\sigma$ is called the *SOR matrix*. Note that $\sigma = 1$ gives $\mathcal{G}$, the Gauss–Seidel matrix.

3.4.4 Convergence of the SOR, Jacobi, and Gauss–Seidel Methods

We present a theorem for convergence of the SOR method, then additional theorems for the Jacobi and Gauss–Seidel methods. Consider first the general iteration equations (3.36) and (3.37) (on page 141). We have

$$x^{(k+1)} - x^{(k)} = B(x^{(k)} - x^{(k-1)}), \quad (3.48)$$

since

$$x^{(k+1)} = Bx^{(k)} + c \quad \text{and} \quad x^{(k)} = Bx^{(k-1)} + c.$$

Also,

$$(I - B)(x^{(k)} - x) = x^{(k)} - x^{(k+1)} \quad (3.49)$$

because

$$x = Bx + c \quad \text{and} \quad x^{(k+1)} = Bx^{(k)} + c.$$

Thus,

$$x^{(k)} - x = -(I - B)^{-1}B(x^{(k)} - x^{(k-1)}) = -(I - B)^{-1}B^2(x^{(k-1)} - x^{(k-2)}) \dots,$$

which leads to the error estimates

$$\|x^{(k)} - x\| \leq \frac{\|B\|}{1 - \|B\|} \|x^{(k)} - x^{(k-1)}\| \quad (3.50)$$

and

$$\|x^{(k)} - x\| \leq \|(I - B)^{-1}\| \|B^k\| \|x^{(1)} - x^{(0)}\| \leq \frac{\|B\|^k}{1 - \|B\|} \|x^{(1)} - x^{(0)}\|, \quad (3.51)$$

assuming, of course, $\|B\| < 1$.

REMARK 3.36 The iteration equation (3.4) (on page 142) corresponds to the iteration equation $x_{t+1} = g(x_t)$ for the fixed point method in the contraction mapping theorem (Theorem 2.3 on page 40), while equations (3.50)

and (3.51) correspond to the error bounds (2.3) and (2.2) in the conclusion to Theorem 2.3, respectively. The proofs of the error estimates (3.50) and (3.51) are completely analogous to the proofs in Theorem 2.3. Further, we will see a nonlinear multidimensional version of the contraction mapping theorem, Theorem 8.2 on page 442, when we study the numerical solution of systems of nonlinear equations. The theorem even generalizes to the solution of integral equations; the proofs are analogous, with norms defined on the function spaces associated with the integral equations.¹⁴ $\square$

REMARK 3.37 These error estimates are only helpful to the extent that a norm $\|\cdot\|$ can be found such that $\|B\| < 1$. Three natural possible norms are

$$\|B\|_\infty = \max_{1 \leq j \leq n} \sum_{k=1}^n |b_{jk}|, \quad \|B\|_1 = \max_{1 \leq k \leq n} \sum_{j=1}^n |b_{jk}|, \quad \text{or} \quad \|B\|_2 = \sqrt{\rho(B^T B)}.$$

In the theorems to follow, we will use these norms (and $\|\cdot\|_2$ in particular.) Also, the proofs sometimes take account of special properties of each method. $\square$

We will now prove a well-known convergence theorem for the SOR method.

THEOREM 3.15

(Ostrowski–Reich) *If A is Hermitian positive definite and $0 < \sigma < 2$, then the SOR method converges for any initial vector $x^{(0)}$.*

We will need the following two lemmas.

LEMMA 3.5

(Stein’s Theorem) *If A is Hermitian positive definite and R is an $n \times n$ matrix such that $A - R^H A R$ is positive definite, then $\rho(R) < 1$.*

PROOF Let λ be an eigenvalue of R and $u \neq 0$ be a corresponding eigenvector. Then $u^H A u$ and $u^H (A - R^H A R) u$ are real and positive. (Recall that $u^H B u > 0$ whenever B is positive definite.) Thus, $u^H A u > u^H R^H A R u = (\lambda u)^H A \lambda u = |\lambda|^2 u^H A u$. Hence, $|\lambda|^2 < 1$. $\square$

LEMMA 3.6

Let A be Hermitian positive definite and suppose that $A = B - C$, with B nonsingular. Further suppose that $B + B^H - A$ is positive definite. Then

¹⁴We illustrate norms on function spaces in Section 4.2, starting on page 191.

$$\rho(B^{-1}C) < 1.$$

PROOF By Stein's Theorem (Lemma 3.5), it is sufficient to show that

$$Q = A - (B^{-1}C)^H A (B^{-1}C)$$

is positive definite. Since $B^{-1}C = I - B^{-1}A$, we have

$$\begin{aligned} Q &= A - (I - B^{-1}A)^H A (I - B^{-1}A) \\ &= A - (I - (B^{-1}A)^H) A (I - B^{-1}A) \\ &= A - A + (B^{-1}A)^H A (B^{-1}A)^H A (B^{-1}A) + A(B^{-1}A) \\ &= (B^{-1}A)^H A (B^{-1}A)^{-1} (B^{-1}A) - (B^{-1}A)^H A (B^{-1}A) + A(B^{-1}A) \\ &= (B^{-1}A)^H [B - A + ((B^{-1}A)^H)^{-1} A] (B^{-1}A) \\ &= (B^{-1}A)^H [B - A + B^H] (B^{-1}A). \end{aligned}$$

But $B - A + B^H$ is positive definite, so Q is positive definite. (Consider $Q = E^H R E$ with R positive definite and E nonsingular. Then

$$y^H Q y = y^H E^H R E y = z^H R z > 0$$

if $z \neq 0$ and hence if $y \neq 0$, since E is nonsingular.) □

With these two lemmas, we now proceed to prove our theorem on convergence of the SOR method.

PROOF (of Theorem 3.15) Note that $A = B - C$ with

$$B = \frac{1}{\sigma}(D + \sigma L) \text{ and } C = \frac{1}{\sigma}[(1 - \sigma)D - \sigma U].$$

(Recall that $A = L + D + U$.) Also,

$$S_\sigma = B^{-1}C = (\sigma L + D)^{-1}[(1 - \sigma)D - \sigma U].$$

We need to show that $\rho(S_\sigma) = \rho(B^{-1}C) < 1$. By Lemma 3.6, if we show that $B + B^H - A$ is positive definite, then we have proved that $\rho(S_\sigma) < 1$. (Note that B is nonsingular because B has all positive diagonal elements. The latter follows from A being positive definite, i.e., $e_j^H A e_j = a_{jj} > 0$.) Consider therefore

$$\begin{aligned} B + B^H - A &= \frac{2}{\sigma}D + L + L^H - L - D - U \\ &= \left(\frac{2}{\sigma} - 1\right)D \text{ (since } L^H = U) \\ &= \frac{1}{\sigma}(2 - \sigma)D. \end{aligned}$$

Since $0 < \sigma < 2$, $B + B^H - A$ is positive definite¹⁵. $\square$

Note: By Theorem 3.15, if A is Hermitian positive definite (or symmetric positive definite), then the SOR method converges for $0 < \sigma < 2$, and, therefore, the Gauss–Seidel method converges. However, the Jacobi method may not be convergent.

Example 3.12

$$A = \begin{pmatrix} 1 & a & a \\ a & 1 & a \\ a & a & 1 \end{pmatrix}.$$

A is positive definite for $-\frac{1}{2} < a < 1$. However,

$$J = - \begin{pmatrix} 0 & a & a \\ a & 0 & a \\ a & a & 0 \end{pmatrix},$$

and $\rho(J) = |2a|$. Thus, the Jacobi method is only convergent for $-\frac{1}{2} < a < \frac{1}{2}$. $\square$

We now consider additional convergence results for the Jacobi and Gauss–Seidel methods. We give first result without proof.

THEOREM 3.16

(Stein–Rosenberg theorem) *Let $J = -(D^{-1})(L + U)$ be a nonnegative $n \times n$ iteration matrix for the Jacobi method, and let $\mathcal{G}$ be the associated Gauss–Seidel matrix. Then one and only one of the following are valid.*

- (i) $\rho(J) = \rho(\mathcal{G}) = 0$
- (ii) $0 < \rho(\mathcal{G}) < \rho(J) < 1$
- (iii) $1 = \rho(J) = \rho(\mathcal{G})$
- (iv) $1 < \rho(J) < \rho(\mathcal{G})$

Thus, in this case, the Jacobi and Gauss–Seidel methods are either both convergent or both divergent. If convergent, the Gauss–Seidel method is generally faster ($0 < \rho(\mathcal{G}) < \rho(J) < 1$). For a proof of this result, see [96].

¹⁵Note that $x^H D x = \sum_{i=1}^n d_{ii} |x_i|^2 > 0$ if $x \neq 0$.

Example 3.13

Let $A = \begin{pmatrix} -2 & 2 \\ 3 & -2 \end{pmatrix}$ and $J = \begin{pmatrix} 0 & 1 \\ 3/2 & 0 \end{pmatrix}$. Since $\rho(J) = \sqrt{3/2} > 1$, the Gauss–Seidel and Jacobi methods do not converge. $\square$

The following define special classes of matrices related to convergence of iterative methods.

DEFINITION 3.33 An $n \times n$ complex matrix $A = (a_{ij})$ is said to be diagonally dominant if

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}| = \mu_i \text{ for } 1 \leq i \leq n. \quad (3.52)$$

If for all $i, 1 \leq i \leq n$, the above inequality is strict, then matrix A is said to be strictly diagonally dominant.

DEFINITION 3.34 For $n \geq 2$, an $n \times n$ complex matrix A is called reducible if there exists a permutation matrix P such that

$$PAP^{-1} = \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{pmatrix},$$

where A_{11} is an $r \times r$ submatrix, $1 \leq r < n$, and A_{22} is an $(n-r) \times (n-r)$ submatrix. If no such permutation matrix exists, A is called irreducible.

By Lemma 3.10 to be given later, a matrix A is irreducible if and only if its directed graph is strongly connected. Lemma 3.10 is generally much easier to use than Definition 3.34 in determining if a particular matrix is irreducible.

DEFINITION 3.35 We say that a matrix A is irreducibly diagonally dominant if A is irreducible, A is diagonally dominant, and at least one of the inequalities in (3.52) is strict.

THEOREM 3.17

Let $A = (a_{ij})$ be an $n \times n$ complex matrix which is either strictly or irreducibly diagonally dominant. Then the Jacobi and Gauss–Seidel methods are convergent.

To prove this, we need some additional results from matrix theory.

LEMMA 3.7

(Perron–Frobenius) Let $B \geq 0$ be an $n \times n$ irreducible matrix. Then

- (i) B has a positive real eigenvalue equal to its spectral radius;
- (ii) there is an eigenvector $x \geq 0$ corresponding to $\lambda = \rho(B)$.

Lemma 3.7 is a classic result from matrix theory.

LEMMA 3.8

Let A and B be $n \times n$ matrices with $0 \leq |B| \leq A$. Then $\rho(B) \leq \rho(A)$.

(Here $|B| = \{|b_{ij}|\}$, where $B = \{b_{ij}\}$.)

PROOF Let $\sigma = \rho(A)$. For any $\epsilon > 0$, set

$$B_1 = (\sigma + \epsilon)^{-1} B$$

and

$$A_1 = (\sigma + \epsilon)^{-1} A.$$

Then $|B_1| \leq A_1$ and $\rho(A_1) < 1$, so

$$0 \leq |B_1|^k \leq A_1^k \rightarrow 0$$

as $k \rightarrow \infty$. Thus, $\rho(B_1) < 1$, so $\rho(B) < \sigma + \epsilon$. Since ϵ was arbitrary, $\rho(B) \leq \rho(A)$. $\square$

LEMMA 3.9

Let A be strictly or irreducibly diagonally dominant. Then A is nonsingular.

PROOF (partial): Let $A = D - B$ be the splitting of A into its diagonal and off-diagonal parts. If A is strictly diagonally dominant, then D is nonsingular, and if $C = D^{-1}B$, then $\rho(C) \leq \|C\|_\infty < 1$. Thus $I - C = D^{-1}A$ is nonsingular, so A is nonsingular. For a proof of the nonsingularity of an irreducible diagonally dominant matrix, see [68]. $\square$

We will now prove Theorem 3.17.

PROOF (of Theorem 3.17) By Lemma 3.9, A is invertible. Since A is strictly or irreducibly diagonally dominant, we also must have $a_{ii} \neq 0$, $1 \leq i \leq n$. (Otherwise, one or more rows of A would be zero and A would not be invertible.) The elements of the iteration matrix $J = -D^{-1}(L+U) = (B_{ij})$ for the Jacobi method are given by:

$$b_{ij} = \begin{cases} 0 & \text{if } i = j, \\ -\frac{a_{ij}}{a_{ii}} & \text{if } i \neq j. \end{cases} \quad (3.53)$$

If A is strictly diagonally dominant, Definition 3.33 gives

$$\sum_{j=1}^n |b_{ij}| < 1, \quad 1 \leq i \leq n. \quad (3.54)$$

Hence, $\|J\|_\infty < 1$ and the Jacobi method converges.

If A is irreducibly diagonally dominant, Definition 3.35 gives

$$\sum_{j=1}^n |b_{ij}| \leq 1, \quad 1 \leq i \leq n \quad \text{and} \quad \sum_{j=1}^n |b_{kj}| < 1 \quad \text{for some } k. \quad (3.55)$$

Now consider the nonnegative matrix $|B| = (|b_{ij}|)$, and let $\lambda > 0$ be its eigenvalue equal to $\rho(|B|)$, whose existence is guaranteed by Lemma 3.7 (the Perron–Frobenius theorem). Let $x > 0$ be a positive eigenvector associated with λ , and let x be normalized so that $x_p = \max_{1 \leq i \leq n} x_i = 1$. We then have

$$(|B|x)_p = \sum_{j=1}^n |b_{pj}|x_j = \lambda x_p = \lambda, \quad \text{so} \quad \rho(|B|) = \lambda = \sum_{j=1}^n |b_{pj}|x_j.$$

Let $S = \{m_1, m_2, \dots, m_s\}$ be such that $x_{m_\ell} = 1$ for $\ell = 1, 2, \dots, s$, and choose $T = \{q_1, q_2, \dots, q_t\}$ such that $x_{q_r} < 1$ for $r = 1, 2, \dots, t$. We now have two cases to consider:

1. Suppose T is nonempty. Clearly, $t + s = n$ and $S \cup T = \{1, 2, \dots, n\}$. Suppose that $|b_{m_\ell q_r}| = 0$ for $\ell = 1, 2, \dots, s$ and $r = 1, 2, \dots, t$. Then by Lemma 3.10 (on page 157 below), the iteration matrix for the Jacobi method is reducible, and the same subsets apply to matrix A , implying that A is reducible. Thus, for some r and ℓ , $|b_{m_\ell q_r}| \neq 0$. We have then, letting $p = m_\ell$,

$$\rho(|B|) = \sum_{j=1}^n |b_{m_\ell j}|x_j < \sum_{j=1}^n |b_{m_\ell j}| \leq 1 \quad (\text{since } |b_{m_\ell q_r}|x_{q_r} < |b_{m_\ell q_r}|).$$

Thus, $\rho(|B|) < 1$.

2. T is empty. Thus, $S = \{1, 2, \dots, n\}$. Choose $p = k$, so

$$\rho(|B|) = \sum_{j=1}^n |b_{kj}|x_j = \sum_{j=1}^n |b_{kj}| < 1.$$

Thus, $\rho(|B|) < 1$.

Now, Lemma 3.8 yields $\rho(J) \leq \rho(|B|)$, and thus, $\rho(J) < 1$.

For the Gauss–Seidel method, we have $\mathcal{G} = (I - G)^{-1}R$, where $G = -D^{-1}L$ and $R = -D^{-1}U$. Since G is strictly lower triangular, it is easily shown that

$G^n = 0$. Therefore, $\mathcal{G} = (I + G + G^2 + \cdots + G^{n-1})R$. Now it is also easily shown that $|MN| \leq |M||N|$. Thus,

$$|\mathcal{G}| \leq (I + |G| + |G|^2 + \cdots + |G|^{n-1})|R| = (I - |G|)^{-1}|R|.$$

Then, by Lemma 3.8, $\rho(\mathcal{G}) \leq \rho((I - |G|)^{-1}|R|)$. But $(I - |G|)^{-1}|R|$ is the Gauss Seidel iteration matrix associated with the nonnegative Jacobi iteration matrix $|B| = |G| + |R|$. Therefore, from Lemma 3.8 and Theorem 3.16, $0 \leq \rho(\mathcal{G}) \leq \rho(|\mathcal{G}|) < \rho(|B|) < 1$. $\square$

Before continuing, it is interesting to note that the Jacobi method may converge where the Gauss–Seidel method diverges. (As indicated by the previous results, such a system is unusual.)

Example 3.14

Consider

$$\begin{aligned} x_1 & & + x_3 &= 0 \\ -x_1 + x_2 & & &= 0 \\ x_1 + 2x_2 - 3x_3 &= 0. \end{aligned}$$

For this system,

$$A = \begin{pmatrix} 1 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 2 & -3 \end{pmatrix}, \quad J = -D^{-1}(L + U) = \begin{pmatrix} 0 & 0 & -1 \\ 1 & 0 & 0 \\ 1/3 & 2/3 & 0 \end{pmatrix},$$

and

$$\mathcal{G} = -(D + L)^{-1}U = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & -1 \\ 0 & 0 & -1 \end{pmatrix}.$$

We have $\rho(\mathcal{G}) = 1$ but $\rho(J) < 1$ (Exercise 42 on page 186 below), so the Jacobi method converges and the Gauss–Seidel method diverges. $\square$

3.4.5 The Interval Gauss–Seidel Method

The interval Gauss–Seidel method is an alternate method¹⁶ for using floating point arithmetic to obtain mathematically rigorous lower and upper bounds to the solution to a system of linear equations. The interval Gauss–Seidel method has several advantages, especially when there are uncertainties in the right-hand-side vector b that are represented in the form of relatively wide intervals $[\underline{b}_i, \bar{b}_i]$, and when there are also uncertainties $[\underline{a}_{ij}, \bar{a}_{ij}]$ in the coefficients of the matrix A . That is, we assume that the matrix is $\mathbf{A} \in \mathbb{IR}^{n \times n}$,

¹⁶to the interval version of Gaussian elimination of Section 3.3.7 on page 130

$\mathbf{b} \in \mathbb{IR}^n$, and we wish to find an interval vector (or “box”) $\mathbf{x}$ that bounds

$$\Sigma(\mathbf{A}, \mathbf{b}) = \{x \mid Ax = b \text{ for some } A \in \mathbf{A} \text{ and some } b \in \mathbf{b}\}, \quad (3.56)$$

where $\mathbb{IR}^{n \times n}$ denotes the set of all n by n matrices whose entries are intervals, $\mathbb{IR}^n$ denotes the set of all n -vectors whose entries are intervals, and $A \in \mathbf{A}$ means that each element of the point matrix A is contained in the corresponding element of the interval matrix $\mathbf{A}$ (and similarly for $b \in \mathbf{b}$).

The interval Gauss–Seidel method is similar to the point Gauss–Seidel method as defined in (3.43) on page 144, except that, for general systems, we almost always precondition. In particular, let $\tilde{\mathbf{A}} = Y\mathbf{A}$ and $\tilde{\mathbf{b}} = Y\mathbf{b}$, where Y is a *preconditioning matrix*. We then have the *preconditioned system*

$$Y\mathbf{A}x = Y\mathbf{b}, \quad \text{i.e. } \tilde{\mathbf{A}}x = \tilde{\mathbf{b}}. \quad (3.57)$$

We have

THEOREM 3.18

(The solution set for the preconditioned system contains the solution set for the original system.) $\Sigma(\mathbf{A}, \mathbf{b}) \subseteq \Sigma(Y\mathbf{A}, Y\mathbf{b}) = \Sigma(\tilde{\mathbf{A}}, \tilde{\mathbf{b}})$.

This theorem is a fairly straightforward consequence of the subdistributivity (Equation (1.9) on page 25) of interval arithmetic. For a proof of this and other facts concerning interval linear systems, see, for example, [62].

Analogously to the noninterval version of Gauss–Seidel iteration (3.43), the interval Gauss–Seidel method is given as

$$\mathbf{x}_i^{(k+1)} \leftarrow \frac{1}{\tilde{a}_{ii}} \left\{ \tilde{b}_i - \sum_{j=1}^{i-1} \tilde{a}_{ij} \mathbf{x}_j^{(k+1)} - \sum_{j=i+1}^n \tilde{a}_{ij} \mathbf{x}_j^{(k)} \right\} \quad (3.58)$$

for $i = 1, 2, \dots, n$, where a sum is interpreted to be absent if its lower index is greater than its upper index, and with $\mathbf{x}_i^{(0)}$ given for $i = 1, 2, \dots, n$.

REMARK 3.38 As with the interval version of Gaussian elimination (Algorithm 3.5 on page 130), a common preconditioner Y for the interval Gauss–Seidel method is the *inverse midpoint matrix* $Y = (\mathbf{m}(\mathbf{A}))^{-1}$, where $\mathbf{m}(\mathbf{A})$ is the matrix whose elements are midpoints of corresponding elements of the interval matrix $\mathbf{A}$. However, when the elements of $\mathbf{A}$ have particularly large widths, specially designed preconditioners¹⁷ may be more appropriate. $\square$

¹⁷See [44, Chapter 3].

REMARK 3.39 Point iterative methods, including the Gauss–Seidel and SOR methods and the conjugate gradient method explained in Section 3.4.10, are usually preconditioned. Note, however, that computing an inverse of a point matrix A leads to $YA \approx I$, where I is the identity matrix, so the system will already have been solved (except for, possibly, iterative refinement). Moreover, such point iterative methods are usually employed for very large systems of equations, with matrices with “0” for many elements. Although the elements that are 0 need not be stored, the inverse generally does not have 0’s in any of its elements [27], so it may be impractical to even store the inverse, let alone compute it.¹⁸ Thus, special approximations are used for these preconditioners.¹⁹ Preconditioners for the point Gauss–Seidel method, conjugate gradient method, etc. are often viewed as operators that increase the separation between the largest eigenvalue of A and the remaining eigenvalues of A , rather than computing an approximate inverse. $\square$

The following theorem tells us that the interval Gauss–Seidel method can be used to prove existence and uniqueness of a solution of a system of linear equations.

THEOREM 3.19

Suppose (3.58) is used, starting with initial interval vector $\mathbf{x}^{(0)}$, and obtaining interval vector $\mathbf{x}^{(k)}$ after a number of iterations. Then, if $\mathbf{x}^{(k)} \subseteq \mathbf{x}^{(0)}$, for each $A \in \mathbf{A}$ and each $b \in \mathbf{b}$, there is an $x \in \mathbf{x}^{(k)}$ such that $Ax = b$.

The proof of Theorem 3.19 can be found in many places, such as in [44] or [62].

Example 3.15

Consider $Ax = b$, where

$$A = \begin{pmatrix} [0.99, 1.01] & [1.99, 2.01] \\ [2.99, 3.01] & [3.99, 4.01] \end{pmatrix}, \quad b = \begin{pmatrix} [-1.01, -0.99] \\ [0.99, 1.01] \end{pmatrix}, \quad x^{(0)} = \begin{pmatrix} [-10, 10] \\ [-10, 10] \end{pmatrix}.$$

Then,²⁰

$$m(A) = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad Y \approx m(A)^{-1} = \begin{pmatrix} -2.0 & 1.0 \\ 1.5 & -0.5 \end{pmatrix},$$

$$\tilde{A} = YA \subseteq \begin{pmatrix} [0.97, 1.03] & [-0.03, 0.03] \\ [-0.02, 0.02] & [0.98, 1.02] \end{pmatrix}, \quad \tilde{b} = Yb \subseteq \begin{pmatrix} [2.97, 3.03] \\ [-2.02, -1.98] \end{pmatrix}.$$

¹⁸Of course, the inverse could be computed one row at a time, but this may still be impractical for large systems.

¹⁹Much work has appeared in the research literature on such preconditioners

²⁰These computations were done with the aid of INTLAB, a MATLAB toolbox available free of charge for non-commercial use.

We then have

$$\begin{aligned} \mathbf{x}_1^{(1)} &\leftarrow \frac{1}{[0.97, 1.03]} ([2.97, 3.03] - [-0.03, 0.03] [-10, 10]) \\ &\subseteq [2.5922, 3.4330], \\ \mathbf{x}_2^{(1)} &\leftarrow \frac{1}{[0.98, 1.02]} ([-2.02, -1.98] - [-0.02, 0.02] [2.5922, 3.4330]) \\ &\subseteq [-2.1313, -1.8738]. \end{aligned}$$

If we continue this process, we eventually obtain

$$\mathbf{x}^{(4)} = ([2.8215, 3.1895], [-2.1264, -1.8786])^T,$$

which, to four significant figures, is the same as $\mathbf{x}^{(3)}$. Thus, we have found mathematically rigorous bounds on the set of all solutions to $Ax = b$ such that $A \in \mathbf{A}$ and $b \in \mathbf{b}$. $\square$

Note: In Example 3.15, uncertainties of ± 0.01 are present in each element of the matrix and right-hand-side vector. Although the bounds produced with the preconditioned interval Gauss–Seidel method are not guaranteed to be the tightest possible with these uncertainties, they will be closer to the tightest possible when the uncertainties are smaller.

Convergence of the interval Gauss–Seidel method is related closely to convergence of the point Gauss–Seidel method, through the concept of diagonal dominance. We give a hint of this convergence theory here.

DEFINITION 3.36 If $\mathbf{a} = [\underline{a}, \overline{a}]$ is an interval, then the magnitude of $\mathbf{a}$ is defined to be

$$\text{mag}(\mathbf{a}) = \max\{|\underline{a}|, |\overline{a}|\}.$$

Similarly, the mignitude of $\mathbf{a}$ is defined to be

$$\text{mig}(\mathbf{a}) = \min_{a \in \mathbf{a}} |a|.$$

Given the matrix $\tilde{\mathbf{A}}$, form the matrix $H = (h_{ij})$ such that

$$h_{ij} = \begin{cases} \text{mag}(\mathbf{a}_{ij}) & \text{if } i \neq j, \\ \text{mig}(\mathbf{a}_{ij}) & \text{if } i = j. \end{cases}$$

Then, basically, the interval Gauss–Seidel method will be convergent if H is diagonally dominant.

For a careful review of convergence theory for the interval Gauss–Seidel method and other interval methods for linear systems, see [62]. Also, see [76].

3.4.6 Graph–Theoretic Properties of Matrices

Many large matrices occurring in practice are such that most of their entries are 0; such matrices are called *sparse* matrices. To make analysis of systems involving sparse matrices practical, we introduce some graph-theoretic concepts. We will also use these concepts in our analysis of the SOR method. Finally, we have already used Lemma 3.10 in this section in the proof of Theorem 3.17 (diagonal dominance implies convergence of the Jacobi and Gauss–Seidel methods).

DEFINITION 3.37 Consider n distinct points $P_1, P_2, \dots, P_n$, which we will call nodes, and consider a matrix $A \in L(\mathbb{R}^n)$. For every nonzero element a_{ij} of A we construct a directed path from P_i to P_j . Thus, we associate with matrix A a directed graph. We will call this the graph of the matrix.

DEFINITION 3.38 We say a directed graph is strongly connected if, for any pair of nodes P_i, P_j there is a path $P_i P_{\ell_1}, P_{\ell_1} P_{\ell_2}, \dots, P_{\ell_m} P_j$ connecting P_i and P_j .

LEMMA 3.10

A matrix A is irreducible if and only if its directed graph is strongly connected.

(For a proof, see [68]).

Example 3.16

Let $A = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$, $B = \begin{pmatrix} 0 & 1 & 0 \\ 3 & 0 & 4 \\ 0 & 2 & 0 \end{pmatrix}$, and $C = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 3 & 0 \\ 5 & 1 & 0 \end{pmatrix}$. Then A

and B are irreducible, but C is reducible. (See Figure 3.1, and also Exercise 44 on page 186.) $\square$

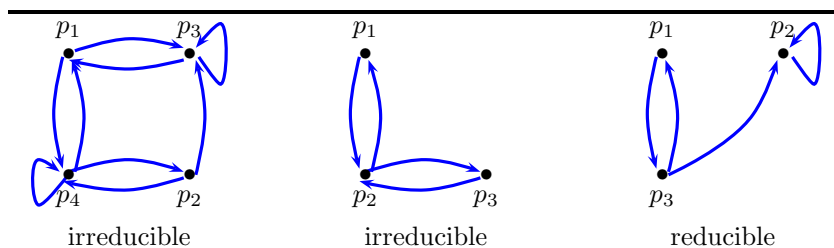


FIGURE 3.1: Directed graphs for irreducible and reducible matrices.

LEMMA 3.11

An $n \times n$ matrix $A = (a_{ij})$ is reducible if and only if there exist two nonempty disjoint subsets S and T of $\{1, 2, \dots, n\}$ such that $S \cup T = \{1, 2, \dots, n\}$ and such that if $i \in S$ and $j \in T$ then $a_{ij} = 0$.

Finally, to state conditions in the theory that follows, we need the following two definitions.

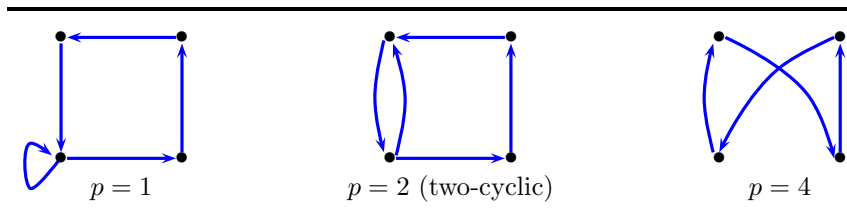


FIGURE 3.2: Cyclic diagrams.

DEFINITION 3.39 The matrix A has property A (or is two-cyclic) if there is a permutation matrix P such that

$$PAP^T = \begin{pmatrix} D_1 & C_1 \\ C_2 & D_2 \end{pmatrix},$$

where D_1 and D_2 are diagonal. Similarly, we say a graph (or matrix) is cyclic of index p if its directed graph is strongly connected and the greatest common divisor of all lengths of its closed paths is p . (See Figure 3.2.)

REMARK 3.40 It can be shown that a matrix A is two-cyclic if its directed graph of its associated iteration matrix for the Jacobi method J is a cyclic graph of index 2, i.e., the greatest common divisor of all lengths of its closed paths is 2. $\square$

DEFINITION 3.40 A matrix is consistently ordered provided the vertices of its adjacency graph can be partitioned into p sets $S_1, S_2, \dots, S_p$ such that any two adjacent vertices P_i and P_j belong to two consecutive partitions S_k and $S_{k'}$, with $k' = k-1$ if $j < i$ and $k' = k+1$ if $j > i$. (An equivalent definition is that the eigenvalues of $B(\alpha) = \alpha^{-1}D^{-1}L + \alpha D^{-1}U$ are independent of α for $\alpha \neq 0$.)

3.4.7 Convergence Rate of the SOR Method

We now have the following question: Can we find an optimum value for σ in the SOR method? The following theorem indicates that we can.

THEOREM 3.20

Let A be a real $n \times n$ matrix which has property A (that is, A is two-cyclic), is consistently ordered, and has nonzero diagonal elements. In addition, assume that the eigenvalues of the iteration matrix for the Jacobi method are real and $\rho(J) < 1$. Then for any $\sigma \in (0, 2)$, $\rho(\mathcal{S}_\sigma) < 1$, i.e., the SOR method converges. Moreover, there is a unique value of $\sigma = \sigma_0$, given by

$$\sigma_0 = \frac{2}{1 + \sqrt{1 - (\rho(J))^2}},$$

for which $\rho(\mathcal{S}_{\sigma_0}) = \min_{0 < \sigma < 2} \rho(\mathcal{S}_\sigma)$ and $\rho(\mathcal{S}_{\sigma_0}) = \sigma_0 - 1$, i.e., for which the SOR method has optimal asymptotic rate of convergence.

For a proof, see [68].

REMARK 3.41 It can be shown that if A is positive definite and tridiagonal then A satisfies the conditions of Theorem 3.20, i.e., A has property A , A is consistently ordered, A has nonzero diagonal elements, and J has real eigenvalues with $\rho(J) < 1$. $\square$

Example 3.17

Let

$$A = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \quad \text{and} \quad J = - \begin{pmatrix} 0 & 1/2 & 0 \\ 1/2 & 0 & 1/2 \\ 0 & 1/2 & 0 \end{pmatrix},$$

so

$$\lambda_1 = 0, \quad \lambda_2 = \frac{1}{\sqrt{2}}, \quad \lambda_3 = -\frac{1}{\sqrt{2}}.$$

Thus,

$$\rho(J) = \frac{1}{\sqrt{2}} \approx 0.7071.$$

By Theorem 3.20, $\sigma_0 = 2/(1 + \sqrt{1 - \rho^2(J)}) = 4 - 2\sqrt{2} \approx 1.1716$, and $\rho(\mathcal{S}_{\sigma_0}) = \sigma_0 - 1 = 0.1716$. In addition,

$$\begin{aligned} \mathcal{S}_{\sigma_1} = \mathcal{G} &= -(L + D)^{-1}U \\ &= \begin{pmatrix} 0 & -1/2 & 0 \\ 0 & 1/4 & -1/2 \\ 0 & -1/8 & 1/4 \end{pmatrix}, \end{aligned}$$

with

$$\lambda_1 = 0, \quad \lambda_2 = 0, \quad \lambda_3 = \frac{1}{2}.$$

Thus, $\rho(\mathcal{S}_{\sigma_1}) = 1/2$. Note that $0 < \underset{0.1716}{\rho(\mathcal{S}_{\sigma_0})} < \underset{0.5000}{\rho(\mathcal{S}_{\sigma_1})} < \underset{0.7071}{\rho(J)} < 1$. □

3.4.8 Convergence of General Matrix Fixed Point Iterations

There is one more useful concept. Recall that

$$x^{(k+1)} = Bx^{(k)} + c, \quad k = 0, 1, 2, \dots \quad (3.36)$$

is the general iterative method considered, and that the error in the k -th iterate is given as

$$\epsilon^{(k+1)} = B^k \epsilon^{(0)}, \text{ where } \epsilon^{(k)} = x^{(k)} - x. \quad (3.37)$$

Thus, how small $\|B^k\|$ is determines how fast $\epsilon^{(k)} \rightarrow 0$, since

$$\|\epsilon^{(k+1)}\| \leq \|B^k\| \|\epsilon^{(0)}\|.$$

The following result is relevant to the size of $\|B^k\|$.

THEOREM 3.21

For any $n \times n$ matrix B and any matrix norm $\|\cdot\|$,

$$\lim_{k \rightarrow \infty} \|B^k\|^{1/k} = \rho(B).$$

The proof of Theorem 3.21 follows from arguments in Section 5.2 (starting on page 297; see Exercise 5 on page 320). We also give an alternate proof, after an additional remark and definition.

REMARK 3.42 From (3.37), we have $\|\epsilon^{(k+1)}\| \leq \|B^k\| \|\epsilon^{(0)}\|$. But for sufficiently large k , $\|B^k\| \approx \rho^k(B)$ so $\|\epsilon^{(k+1)}\|$ is approximately less than or equal to $\rho^k(B) \|\epsilon^{(0)}\|$. We can therefore compare the number of iterations expected to reduce the error below a certain tolerance τ for two different methods. Suppose that iteration matrices B_1 and B_2 are different for two methods. Suppose that $\|\epsilon^{(k_1)}\| \leq \rho^{k_1}(B_1) \|\epsilon^{(0)}\| = \tau$ and $\|\epsilon^{(k_2)}\| \leq \rho^{k_2}(B_2) \|\epsilon^{(0)}\| = \tau$. Thus, to ensure that the error is below tolerance τ , $\rho^{k_1}(B_1) = \rho^{k_2}(B_2)$, whence

$$\frac{k_1}{k_2} = \frac{\ln \rho(B_2)}{\ln \rho(B_1)}.$$

Hence, the convergence rate is proportional to $-\ln \rho(B)$, since the number of iterations to convergence is proportional to $-1/\ln \rho(B)$. □

Remark 3.42 leads us to

DEFINITION 3.41 *Let B be a convergent iteration matrix ($\rho(B) < 1$). The asymptotic rate of convergence of the iteration is defined by*

$$R_\infty(B) = -\ln \rho(B).$$

PROOF (of Theorem 3.21) Recall that $\rho(B) \leq \|B\|$ for any matrix norm, whence $[\rho(B)]^k = \rho(B^k) \leq \|B^k\|$, that is,

$$\rho(B) \leq \|B^k\|^{1/k} \quad \text{for } k = 1, 2, \dots \quad (3.59)$$

Now let $\epsilon > 0$ be arbitrary. We define

$$B(\epsilon) = \frac{B}{\rho(B) + \epsilon}, \quad \text{so } \rho(B(\epsilon)) = \frac{\rho(B)}{\rho(B) + \epsilon} < 1.$$

This in turn gives $\lim_{k \rightarrow \infty} (B(\epsilon))^k = 0$, that is, $\lim_{k \rightarrow \infty} \|(B(\epsilon))^k\| = 0$. Hence, there exists a k_0 , depending on ϵ , such that

$$\|(B(\epsilon))^k\| = \frac{\|B^k\|}{(\rho(B) + \epsilon)^k} < 1 \quad \text{for } k \geq k_0.$$

Thus,

$$\|B^k\|^{1/k} \leq \rho(B) + \epsilon \quad \text{for } k \geq k_0.$$

Taking the limit of the left member then gives

$$\lim_{k \rightarrow \infty} \|B^k\|^{1/k} \leq \rho(B) + \epsilon.$$

for $\epsilon > 0$. Since ϵ was arbitrary, $\lim_{k \rightarrow \infty} \|B^k\|^{1/k} \leq \rho(B)$. Combining this result with (3.59) gives $\lim_{k \rightarrow \infty} \|B^k\|^{1/k} = \rho(B)$. $\square$

3.4.9 The Block SOR Method

We now consider briefly the *Block SOR* iterative method before considering in detail the conjugate gradient method. Consider

$$A = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1\nu} \\ A_{21} & \ddots & & \vdots \\ \vdots & & & \\ A_{\nu 1} & \cdots & & A_{\nu\nu} \end{pmatrix}, \quad (3.60)$$

where the A_{ii} , $1 \leq i \leq \nu$ are square $r_i \times r_i$ matrices with $r_i \geq 1$, $1 \leq i \leq \nu$, and

$$\sum_{i=1}^{\nu} r_i = n,$$

where the entire matrix A is n by n . Corresponding to the partitioning (3.60), we define a block diagonal matrix D , a block lower triangular matrix L , and a block upper triangular matrix U by

$$D = \begin{pmatrix} A_{11} & 0 & \cdots & 0 \\ 0 & A_{22} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & A_{\nu\nu} \end{pmatrix}, \quad (3.61)$$

$$L = \begin{pmatrix} 0 & \cdots & 0 \\ A_{21} & 0 & \vdots \\ \vdots & \ddots & \\ A_{\nu 1} & \cdots & A_{\nu, \nu-1} & 0 \end{pmatrix}, \quad U = \begin{pmatrix} 0 & A_{12} & \cdots & A_{1\nu} \\ & & \ddots & \vdots \\ \vdots & & & A_{\nu-1, \nu} \\ 0 & \cdots & & 0 \end{pmatrix}.$$

Assume that D is nonsingular. Then $(D + \sigma L)$ is invertible for all σ . If we partition x and b consistently with (3.60), then $Ax = b$ can be written as

$$\begin{pmatrix} A_{11} & \cdots & A_{1\nu} \\ A_{21} & \cdots & A_{2\nu} \\ \vdots & & \\ A_{\nu 1} & \cdots & A_{\nu\nu} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_\nu \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_\nu \end{pmatrix}.$$

That is, $\sum_{j=1}^{\nu} A_{ij}x_j = b_i$, $1 \leq i \leq \nu$. We now obtain the block SOR method defined by

$$A_{ii}x_i^{(k+1)} = (1 - \sigma)A_{ii}x_i^{(k)} + \sigma \left\{ -\sum_{j=1}^{i-1} A_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{\nu} A_{ij}x_j^{(k)} + b_i \right\} \quad (3.62)$$

for $k = 0, 1, 2, \dots$ and $1 \leq i \leq \nu$, with $x^{(0)}$ given.

REMARK 3.43 If $\sigma = 1$, we obtain the block Gauss–Seidel method. $\square$

REMARK 3.44 If, for example, D is positive definite and A is positive definite, then the block SOR method converges for $0 < \sigma < 2$. $\square$

Block methods allow us to efficiently handle systems in which the pattern of nonzeros is in a block structure, such as in discretizations of various types of partial differential equations. Block methods can lead to efficient use of resources in parallel computing, etc.

3.4.10 The Conjugate Gradient Method

We now consider the conjugate gradient method. It is assumed here that A is symmetric positive definite. This method is well-suited for sparse matrices (many zero elements). Consider the linear system $Ax = b$ and consider the corresponding quadratic functional

$$F(v) = \frac{1}{2} \sum_{i,j=1}^n a_{ij} v_i v_j - \sum_{i=1}^n b_i v_i = \frac{1}{2} (Av, v) - (b, v), \quad (3.63)$$

where $(x, y) = x^T y$. Since A is real symmetric positive definite,

$$(Av, v) > 0 \text{ if } v \neq 0. \quad (3.64)$$

Now consider the derivative of $F(v)$ with respect to v_i . Then

$$\frac{\partial F}{\partial v_i} = \sum_{j=1}^n a_{ij} v_j - b_i. \quad (3.65)$$

First, recall that the *residual vector* r for v is defined by

$$r = Av - b, \quad (3.66)$$

whence

$$\nabla F = r = Av - b. \quad (3.67)$$

This leads to

THEOREM 3.22

The solution to $Ax = b$ (assuming that A is symmetric positive definite) is equivalent to the problem of finding the minimum of the quadratic functional $F(v)$ of (3.63).

PROOF The solution of $Ax = b$ has zero residual vector $r = Ax - b = 0$. But if v minimizes (3.63), a necessary condition is $\nabla F(v) = 0$. Thus, one such v is clearly $v = x$, since $\nabla F(x) = r = 0$. But since A is positive definite, $F(v)$ has one and only one such critical point. In addition, x gives a minimum of $F(v)$. (To see this, consider $F(x + \Delta x) = F(x) + \frac{1}{2}(A\Delta x, \Delta x)$. Thus, $F(x) \leq F(x + \Delta x)$ for any $\Delta x \neq 0$.) Conversely, if x is a minimum of F , then by (3.67), $Ax - b = 0$, or x is the solution of $Ax = b$. $\square$

The conjugate gradient method finds the minimum x of F in at most n steps. In the conjugate gradient method, v is a given vector, and we choose a vector $p \neq 0$. We then let $v' = v + tp$, where t is specified later. For fixed v and specified p , $F(v')$ is a quadratic function of t only. It is given by

$$F(v') = F(v + tp) = \frac{1}{2} (Av, v) + t(Av, p) + \frac{1}{2} t^2 (Ap, p) - (b, v) - t(b, p),$$

or, using (3.63) and (3.66),

$$F(v') = \frac{1}{2}t^2(Ap, p) + t(r, p) + F(v). \quad (3.68)$$

We now choose t so that if we start at v and go in direction p , then v' will minimize F . A necessary condition is $\frac{dF}{dt} = t(Ap, p) + (r, p) = 0$ or

$$\hat{t} = -\frac{(r, p)}{(Ap, p)}. \quad (3.69)$$

Since $\frac{d^2F}{dt^2} = (Ap, p) > 0$, choosing $t = \hat{t}$ produces a minimum of F along the *relaxation direction* p . The point $\hat{v} = v + \hat{t}p$ is called a minimum point.

We now describe the conjugate gradient method procedure.

Step 1. Choose initial vector $v^{(0)}$ and direction

$$p^{(1)} = -r^{(0)} = -\text{grad}(F(v^{(0)})) = -Av^{(0)} + b.$$

Then

$$v^{(1)} = v^{(0)} - t_1 r^{(0)},$$

where

$$t_1 = \frac{(r^{(0)}, r^{(0)})}{(Ar^{(0)}, r^{(0)})} = -\frac{(r^{(0)}, p^{(1)})}{(Ap^{(1)}, p^{(1)})}.$$

Step 2. We are given $v^{(1)}$. We now choose $p^{(2)}$ as a linear combination of $r^{(1)}$ and $p^{(1)}$. ($-r^{(1)} = -Av^{(1)} + b = -\nabla F(v^{(1)})$.) We let

$$p^{(2)} = -r^{(1)} + \mu_1 p^{(1)},$$

where μ_1 is chosen so that $(Ap^{(2)}, p^{(1)}) = (p^{(2)}, Ap^{(1)}) = 0$. Thus,

$$\mu_1 = \frac{(r^{(1)}, Ap^{(1)})}{(p^{(1)}, Ap^{(1)})}.$$

We now proceed along $p^{(2)}$ in the usual manner to a minimum point.

After the k -th step, our directions and residual vectors satisfy the following for, $k = 2, 3, 4, \dots$

$$v^{(k)} = v^{(k-1)} + t_k p^{(k)}, \text{ where } t_k = \frac{-(r^{(k-1)}, p^{(k)})}{(Ap^{(k)}, p^{(k)})}, \quad (3.70)$$

$$r^{(k)} = Av^{(k)} - b = r^{(k-1)} + t_k Ap^{(k)}, \quad (3.71)$$

$$p^{(k)} = -r^{(k-1)} + \mu_{k-1} p^{(k-1)}, \text{ where } \mu_{k-1} = \frac{(r^{(k-1)}, Ap^{(k-1)})}{(p^{(k-1)}, Ap^{(k-1)})}. \quad (3.72)$$

We have the following result:

PROPOSITION 3.11

$(r^{(k)}, r^{(k-1)}) = (r^{(k)}, p^{(k-1)}) = (r^{(k)}, p^{(k)}) = 0$ for $k = 2, 3, \dots$. In addition, $(r^{(1)}, r^{(0)}) = 0$ and $(r^{(1)}, p^{(1)}) = 0$.

PROOF

$$\begin{aligned} -(r^{(1)}, p^{(1)}) &= (r^{(1)}, r^{(0)}) = (Av^{(1)} - b, r^{(0)}) \\ &= (Av^{(0)} - t_1 Ar^{(0)} - b, r^{(0)}) \\ &= (r^{(0)} + b - t_1 Ar^{(0)} - b, r^{(0)}) \\ &= (r^{(0)}, r^{(0)}) - t_1 (Ar^{(0)}, r^{(0)}) = 0. \end{aligned}$$

The proof now proceeds by induction. Consider the k -th step. By (3.71),

$$(r^{(k)}, p^{(k)}) = (r^{(k-1)}, p^{(k)}) + t_k (Ap^{(k)}, p^{(k)}) = 0,$$

since $t_k = -(r^{(k-1)}, p^{(k)}) / (Ap^{(k)}, p^{(k)})$. By (3.71) and (3.72),

$$\begin{aligned} (r^{(k)}, p^{(k-1)}) &= (r^{(k-1)}, p^{(k-1)}) + t_k (Ap^{(k)}, p^{(k-1)}) \\ &= t_k [(-Ar^{(k-1)} + \mu_{k-1} Ap^{(k-1)}, p^{(k-1)})] \\ &= t_k (-Ar^{(k-1)}, p^{(k-1)}) + t_k \mu_{k-1} (Ap^{(k-1)}, p^{(k-1)}) \\ &= 0 \text{ by definition of } \mu_{k-1}. \end{aligned}$$

Finally, by what we just showed and (3.72),

$$0 = (p^{(k)}, r^{(k)}) = -(r^{(k)}, r^{(k-1)}) + \mu_{k-1} (p^{(k-1)}, r^{(k)}).$$

Thus, $(r^{(k)}, r^{(k-1)}) = 0$. □

REMARK 3.45 μ_{k-1} and t_k are well-defined as long as $(Ap^{(k)}, p^{(k)}) \neq 0$. But since A is positive definite $(Ap^{(k)}, p^{(k)}) > 0$ for $p^{(k)} \neq 0$. If $p^{(k)} = 0$, then by (3.72), $r^{(k-1)} = \mu_{k-1} p^{(k-1)}$ and hence since $(r^{(k-1)}, p^{(k-1)}) = 0$, $(r^{(k-1)}, r^{(k-1)}) = 0$, and thus $r^{(k-1)} = 0$. This indicates that $0 = r^{(k-1)} = Av^{(k-1)} - b$ or that $v^{(k-1)}$ is the solution. □

We now have:

THEOREM 3.23

In the conjugate gradient method, the residual vectors $r^{(k)}$, $k = 0, 1, 2, \dots$ form an orthogonal system, i.e., $(r^{(i)}, r^{(j)}) = 0$ for $i \neq j$.

PROOF (By Induction) Assume that after the k -th step we have

$$(r^{(i)}, r^{(j)}) = 0 \text{ for } i \neq j, 0 \leq i, j \leq k \quad (3.73)$$

$$(p^{(i)}, Ap^{(j)}) = 0 \text{ for } i \neq j, 0 \leq i, j \leq k. \quad (3.74)$$

By the induction hypothesis, (3.73) holds for $k = 1$, while (3.74) holds also for $k = 1$, because $p^{(0)}$ can be set equal to 0. We need to prove the following:

$$(p^{(k+1)}, Ap^{(j)}) = 0 \text{ for } j = 1, 2, \dots, k \text{ and} \quad (3.75)$$

$$(r^{(k+1)}, r^{(j)}) = 0 \text{ for } j = 0, 1, 2, \dots, k. \quad (3.76)$$

For $j = k$, (3.75) is satisfied by construction of direction $p^{(k+1)}$ as seen by (3.72). By (3.72) and (3.74), we have for $1 \leq j < k$ that

$$(p^{(k+1)}, Ap^{(j)}) = -(r^{(k)}, Ap^{(j)}) + \mu_k(p^{(k)}, Ap^{(j)}) = -(r^{(k)}, Ap^{(j)}).$$

But by (3.71),

$$Ap^{(j)} = \frac{r^{(j)} - r^{(j-1)}}{t_j}, \text{ so } (p^{(k+1)}, Ap^{(j)}) = -\frac{1}{t_j}[(r^{(k)}, r^{(j)}) - (r^{(k)}, r^{(j-1)})] = 0$$

by (3.73). Thus, (3.75) holds for $1 \leq j \leq k$.

Now consider (3.76). (3.76) holds for $j = k$ by the previous proposition. For $0 \leq j < k$, we note that by (3.71) and (3.73),

$$\begin{aligned} (r^{(k+1)}, r^{(j)}) &= (r^{(k)}, r^{(j)}) + t_{k+1}(Ap^{(k+1)}, r^{(j)}) \\ &= t_{k+1}(Ap^{(k+1)}, r^{(j)}) \end{aligned} \quad (3.77)$$

Using (3.72) with j in place of $k - 1$, we have for $1 \leq j < k$,

$$(r^{(k+1)}, r^{(j)}) = t_{k+1}(-(Ap^{(k+1)}, p^{(j+1)}) + \mu_j(Ap^{(k+1)}, p^{(j)})).$$

By (3.75), which we just proved, we have (3.76) for $1 \leq j \leq k$. Finally, for $j = 0$, $r^{(0)} = -p^{(1)}$ and by (3.71), (3.73), and (3.75),

$$(r^{(k+1)}, r^{(0)}) = t_{k+1}(Ap^{(k+1)}, r^{(0)}) = -t_{k+1}(Ap^{(k+1)}, p^{(1)}) = 0.$$

□

We now have the following remarkable result that the conjugate gradient method is actually a finite procedure.

COROLLARY 3.1

The conjugate gradient method yields the solution of $Ax = b$ in at most n steps.

PROOF By Theorem 3.23, the residual vectors $r^{(k)}, k \geq 0$, form an orthogonal system. But these vectors belong to $\mathbb{R}^n$ and hence contain at

most n nonzero vectors. Thus, by the n -th step, $r^{(n)} = 0$ and thus $v^{(n)}$ satisfies $Av^{(n)} - b = r^{(n)} = 0$, i.e., $v^{(n)}$ is the solution of $Ax = b$. $\square$

REMARK 3.46 Although the solution should be obtained in at most n steps, due to rounding error, the residual vector $r^{(n)}$ may be nonzero. Calculations may therefore be continued beyond the n -th step. Also, especially for n large, the iterations may yield an approximate solution of sufficient accuracy before the n -th step. $\square$

REMARK 3.47 This method is very attractive for sparse positive definite matrices A . The bulk of the work in each step is expended in calculating $Ap^{(k)}$, a matrix-vector product which requires $\mathcal{O}(n^2)$ computations, but this work is much reduced if A is sparse. For example, if each row of A contains α nonzero entries, then $Ap^{(k)}$ requires $\mathcal{O}(\alpha n)$ work. $\square$

3.4.11 Iterative Methods for Matrix Inversion

We now briefly consider iterative methods for matrix inversion. However, in practice, it is usually not necessary to find the inverse of a matrix. Except when the matrix inverse is explicitly required,²¹ it is faster to use a direct method such as Gaussian elimination or an iterative method such as the SOR method to solve a single system of equations. We nonetheless consider two iterative methods for matrix inversion.

THEOREM 3.24

(Iterative method 1) Let A be an $n \times n$ nonsingular matrix. Suppose that C is an $n \times n$ matrix such that $\|I - AC\| \leq q < 1$. Then for any nonsingular matrix X_0 , the sequence

$$X_{k+1} = X_k B + C, \text{ where } B = I - AC \quad (3.78)$$

converges to A^{-1} , with error estimates

$$\|X_k - A^{-1}\| \leq \frac{q}{1-q} \|X_k - X_{k-1}\| \leq \frac{q^k}{1-q} \|X_1 - X_0\|, \quad k = 1, 2, \dots \quad (3.79)$$

PROOF By hypothesis, $\|B\| \leq q < 1$. Thus,

$$(I - B)^{-1} = I + B + \dots + B^n + \dots = \sum_{j=0}^{\infty} B^j.$$

²¹such as for sensitivity analysis or in preconditioning an interval method

Note that $(AC)^{-1} = \sum_{j=0}^{\infty} B^j$. Since A and AC are nonsingular, C is nonsingular.

Now consider $X_k = X_{k-1}B + C$. Thus,

$$X_1 = X_0B + C, X_2 = X_1B + C = X_0B^2 + CB + C$$

and, in general, $X_k = C \sum_{j=0}^{k-1} B^j + X_0B^k$ for $k = 1, 2, \dots$.

Hence, $X = \lim_{k \rightarrow \infty} X_k = C \sum_{j=0}^{\infty} B^j = C(AC)^{-1} = A^{-1}$.

In addition, since $A^{-1} = A^{-1}B + C$, we have $X_{k+1} - A^{-1} = (X_k - A^{-1})B$. Thus, $(X_{k+1} - A^{-1}) - (X_k - A^{-1}) = (X_k - A^{-1})(B - I)$ and hence $(X_k - A^{-1}) = (X_k - X_{k+1})(I - B)^{-1}$. Therefore,

$$\|X_k - A^{-1}\| \leq \frac{\|X_k - X_{k+1}\|}{1 - q}. \quad (3.80)$$

(Recall that $\|(I - B)^{-1}\| \leq \frac{1}{1 - \|B\|} \leq \frac{1}{1 - q}$.) Also, $(X_{k+1} - X_k) = (X_k - X_{k-1})B$ (since $X_{k+1} = BX_k + C$ and $X_k = BX_{k-1} + C$). Hence,

$$\|X_{k+1} - X_k\| \leq q\|X_k - X_{k-1}\|. \quad (3.81)$$

Inequalities (3.80) and (3.81) finally give

$$\|X_k - A^{-1}\| \leq \frac{q}{1 - q}\|X_k - X_{k-1}\| \leq \frac{q^k}{1 - q}\|X_1 - X_0\|.$$

□

THEOREM 3.25

(Iterative method 2 — actually a Newton method) Let A be nonsingular. Suppose that $\|R_0\| = \|I - AX_0\| \leq q < 1$. Then the iterates defined by

$$X_{k+1} = X_k(I + R_k), \quad R_k = I - AX_k, \quad k = 0, 1, 2, \dots \quad (3.82)$$

converge to A^{-1} with error estimates

$$\|X_k - A^{-1}\| \leq \frac{\|X_0\|}{1 - q}\|I - AX_k\| \leq \frac{\|X_0\|q^{2^k}}{1 - q}. \quad (3.83)$$

PROOF With the above notation

$$\begin{aligned} R_k &= I - AX_k = I - AX_{k-1}(I + R_{k-1}) = R_{k-1} - AX_{k-1}R_{k-1} = \\ &= (I - AX_{k-1})R_{k-1} = R_{k-1}^2. \end{aligned} \quad (3.84)$$

Thus,

$$R_k = (R_0)^{2^k} \text{ for } k = 0, 1, 2, \dots \quad (3.85)$$

In addition, as $\|R_0\| < 1$, $\|R_k\| \rightarrow 0$ as $k \rightarrow \infty$ and $X_k = A^{-1}(I - R_k) \rightarrow A^{-1}$ as $k \rightarrow \infty$. Also, $R_0 = I - AX_0$ and thus $A^{-1} = X_0(I - R_0)^{-1}$ gives

$$\|A^{-1}\| \leq \frac{\|X_0\|}{1 - q}. \quad (3.86)$$

Therefore,

$$\begin{aligned} \|X_k - A^{-1}\| &\leq \|A^{-1}\| \|I - AX_k\| = \|A^{-1}\| \|R_k\| \leq \frac{\|X_0\|}{1 - q} \|R_0\|^{2^k} \\ &\leq \frac{\|X_0\|}{1 - q} q^{2^k}. \end{aligned} \quad (3.87)$$

□

3.4.12 Krylov Subspace Methods

We now briefly consider Krylov subspace methods for approximate solution of $Ax = b$, although these methods are also useful for finding eigenvalues (Chapter 5). It is assumed in the following that A is $n \times n$ and nonsingular. References for Krylov subspace methods are [23] or [78]. We begin with

DEFINITION 3.42 *A Krylov subspace is a subspace of the form*

$$K_m(A, v) = \text{span}(v, Av, A^2v, \dots, A^{m-1}v).$$

To motivate Krylov subspace methods, consider the case $y_1 = b$, $y_2 = Ay_1 = Ab$, $\dots$, $y_n = Ay_{n-1} = A^{n-1}y_1 = A^{n-1}b$. Let K be the $n \times n$ matrix

$$K = (y_1, y_2, \dots, y_n).$$

Then $AK = (Ay_1, Ay_2, \dots, Ay_n) = (y_2, y_3, \dots, A^n y_1)$. Assume that K is nonsingular and let $c = -K^{-1}A^n y_1$. Then $AK = K[e_2, e_3, \dots, e_n, -c]$. Thus,

$$K^{-1}AK = C = \begin{pmatrix} 0 & 0 & \cdots & 0 & -c_1 \\ 1 & 0 & \cdots & 0 & -c_2 \\ 0 & 1 & 0 & \cdots & 0 & -c_3 \\ 0 & & & \ddots & \vdots & \vdots \\ \vdots & & & & 0 & \\ 0 & 0 & \cdots & 0 & 1 & -c_n \end{pmatrix} \quad (3.88)$$

(This matrix is called a *companion matrix*.) Note that C is upper Hessenberg²² and C has the same eigenvalues as A because, if $Ax = \lambda x$, then $Cy = \lambda y$ where $y = K^{-1}x$. Also, the characteristic polynomial of C is very simple:

$$p(x) = -\det(C - xI) = x^n + \sum_{i=1}^n c_i x^{i-1}.$$

Thus, the eigenvalues of A can be found by finding the zeros of $p(x)$.

Unfortunately, this technique is not useful in practice, since finding c requires computing $A^n y_1$ and then solving $Kc = A^n y_1$. The matrix K is likely to be ill-conditioned so c would be inaccurately computed.²³

To overcome these problems, we will replace K by an orthogonal matrix Q , such that for all m , the leading m columns of K and Q span the same space. In contrast to K , Q is well-conditioned and easy to invert. Furthermore, we may compute only as many columns as necessary of Q to get an accurate solution. Generally, less columns are required than matrix dimension n .

Thus, let

$$K = QR \text{ (the } QR \text{ decomposition of } K\text{)}.$$

Then $K^{-1}AK = R^{-1}Q^T AQR = C$ implying that

$$Q^T A Q = R C R^{-1} = H.$$

Since R is upper triangular and C is upper Hessenberg, it follows that H is upper Hessenberg. Hence, $Q^T A Q = H$ where Q is unitary and H upper Hessenberg.

REMARK 3.48 If A is real symmetric, then H is also symmetric, so H must be tridiagonal. In this case, we write $Q^T A Q = T$, where T is tridiagonal. $\square$

Now consider computation of Q . Let $Q = (q_1, q_2, \dots, q_n)$. Since $Q^T A Q = H$, $AQ = QH$, and we equate column j on both sides to obtain $Aq_j = \sum_{i=1}^{j+1} h_{i,j} q_i$. Hence,

$$q_m^T A q_j = \sum_{i=1}^{j+1} h_{i,j} q_m^T q_i = h_{m,j} \text{ for } 1 \leq m \leq j \quad (3.89)$$

²²An upper Hessenberg matrix is a matrix $A = [a_{ij}]$ that has nonzeros only on and above the diagonal and in entries immediately below the diagonal; that is, for an upper Hessenberg matrix, $a_{ij} \neq 0$ only if $j \geq i - 1$.

²³The columns of K can be related to successive iterates in the power method (see Section 5.2 on page 297), so y_j and y_{j+1} are likely to be close for j large.

and

$$h_{j+1,j}q_{j+1} = Aq_j - \sum_{i=1}^j h_{i,j}q_i. \quad (3.90)$$

By (3.89) and (3.90), we obtain the following algorithm for finding Q and H .

ALGORITHM 3.7

(Arnoldi's Algorithm)

INPUT: A , b , the number of desired columns k , and a zero tolerance ϵ .

OUTPUT: the first k columns of Q and H .

1. $q_1 \leftarrow b/\|b\|_2$.
2. FOR $j = 1$ to k
 - (a) $z \leftarrow Aq_j$.
 $w \leftarrow z$.
 - (b) FOR $i = 1$ to j
 - i. $h_{i,j} \leftarrow q_i^T w$.
 - ii. $z \leftarrow z - h_{i,j}q_i$.
 - END FOR
 - (c) $h_{j+1,j} \leftarrow \|z\|_2$.
 - (d) IF $|h_{j+1,j}| < \epsilon$ THEN RETURN.
 - (e) $q_{j+1} \leftarrow z/h_{j+1,j}$.

END FOR

END ALGORITHM 3.7.

REMARK 3.49 Arnoldi's algorithm is also called a *modified Gram-Schmidt* algorithm. Components in directions q_1 to q_j are subtracted from z , leaving z orthogonal to them. $\square$

REMARK 3.50 Suppose we use Arnoldi's algorithm to compute k columns of Q . Let $Q = (Q_k, Q_p)$ where $Q_k = (q_1, q_2, \dots, q_k)$ and $Q_p = (q_{k+1}, \dots, q_n)$ and where Q_p is unknown. $\square$

Then

$$\begin{aligned} H &= Q^T A Q = (Q_k, Q_p)^T A (Q_k, Q_p) = \begin{pmatrix} Q_k^T A Q_k & Q_k^T A Q_p \\ Q_p^T A Q_k & Q_p^T A Q_p \end{pmatrix} \\ &= \begin{pmatrix} H_k & H_{pk} \\ H_{kp} & H_p \end{pmatrix} \begin{matrix} k & n-k \\ k & n-k \end{matrix} \end{aligned} \quad (3.91)$$

Note that H_k is upper Hessenberg and H_{kp} has a single (possibly) nonzero entry in its upper right corner, namely, $h_{k+1,k}$. H_k and H_{kp} are known but H_{pk} and H_p are unknown.

Also, note that $A(Q_k, Q_p) = (Q_k, Q_p)H$, so

$$AQ_k = Q_k H_k + Q_p H_{kp} = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T. \quad (3.92)$$

REMARK 3.51 Suppose that A is symmetric. Then $H = T$ is symmetric and tridiagonal.

$$\text{Let } T = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 \\ \beta_1 & \alpha_2 & \beta_2 & \cdots & 0 \\ 0 & & \ddots & & \vdots \\ \vdots & & & & \beta_{n-1} \\ 0 & \cdots & 0 & \beta_{n-1} & \alpha_n \end{pmatrix} \quad (3.93)$$

Equating column j on both sides of $AQ = QT$ yields

$$Aq_j = \beta_{j-1} q_{j-1} + \alpha_j q_j + \beta_j q_{j+1}.$$

In addition, $q_j^T Aq_j = \alpha_j$. This leads to the Lanczos algorithm for A real symmetric. $\square$

ALGORITHM 3.8

(Lanczos Algorithm)

INPUT: A , b , and the number of desired columns k , and a zero tolerance ϵ .

OUTPUT: the first k columns of Q and T .

1. $q_1 \leftarrow b/\|b\|_2$, $\beta_0 \leftarrow 0$, $q_0 \leftarrow 0$.

2. FOR $j = 1$ to k

(a) $z \leftarrow Aq_j$.

(b) $\alpha_j \leftarrow q_j^T z$.

(c) $z \leftarrow z - \alpha_j q_j - \beta_{j-1} q_{j-1}$.

(d) $\beta_j \leftarrow \|z\|_2$.

(e) IF $|\beta_j| < \epsilon$ THEN RETURN.

(f) $q_{j+1} \leftarrow z/\beta_j$.

END FOR

END ALGORITHM 3.8.

After k steps of the Lanczos algorithm, we have

$$T = \begin{pmatrix} T_k & T_{pk}^T \\ T_{pk} & T_p \end{pmatrix} = Q^T A Q = \begin{matrix} k & n-k \\ \begin{pmatrix} Q_k^T A Q_k & Q_k^T A Q_p \\ Q_p^T A Q_k & Q_p^T A Q_p \end{pmatrix} \end{matrix}. \quad (3.94)$$

Because A is symmetric, we know T_k and T_{pk} and $T_{kp} = T_{pk}^T$, but we don't know T_p . T_{pk} has a single (possibly) nonzero element in the upper right corner, β_k .

Now consider approximate solution of $Ax = b$ using Krylov subspace techniques. Consider $K_m(A, r_0) = \text{span}(r_0, Ar_0, \dots, A^{m-1}r_0)$, where $r_0 = b - Ax_0$. $Ax = b$ is solved approximately by seeking a solution $x_m = x_0 + w_m$, where $w_m \in K_m(A, r_0)$ and $b - Ax_m \perp K_m(A, r_0)$.

Let $q_1 = r_0/\|r_0\|_2$ be the starting vector in Arnoldi's algorithm and $\beta = \|r_0\|_2$. Then

$$Q_m^T A Q_m = H_m \text{ and } Q_m^T r_0 = Q_m^T (\beta q_1) = \beta e_1.$$

Then

$$\begin{cases} x_m = x_0 + Q_m y_m \\ y_m = H_m^{-1}(\beta e_1) \text{ or } H_m y_m = \beta e_1. \end{cases} \quad (3.95)$$

This method is called the Full Orthogonalization Method (FOM). To verify that $b - Ax_m \perp K_m(A, r_0)$ consider the following:

$$\begin{aligned} \text{We have } Ax_m - b &= Ax_0 - b + A Q_m y_m \\ &= -r_0 + A Q_m H_m^{-1}(\beta e_1) \end{aligned}$$

$$\begin{aligned} \text{Thus, } Q_m^T [Ax_m - b] &= Q_m^T [-r_0 + A Q_m H_m^{-1}(\beta e_1)] \\ &= -Q_m^T r_0 + \beta e_1 = 0. \end{aligned}$$

$$\text{Hence, } Ax_m - b \perp K_m.$$

We also have the following error result:

PROPOSITION 3.12

The residual vector $b - Ax_m$ computed using the FOM satisfies $\|b - Ax_m\|_2 = |h_{m+1,m}| |e_m^T y_m|$.

PROOF

$$\begin{aligned} \text{We have } b - Ax_m &= b - Ax_0 - A Q_m y_m \\ &= \beta q_1 - Q_m H_m y_m - h_{m+1,m} q_{m+1} e_m^T y_m \\ &= \beta q_1 - \beta q_1 - h_{m+1,m} e_m^T y_m q_{m+1} \end{aligned}$$

Thus, $\|b - Ax_m\|_2 = |h_{m+1,m}| |e_m^T y_m|$. □

REMARK 3.52 The conjugate gradient method (for symmetric positive definite linear systems) is an orthogonal projection technique onto the Krylov subspace $K_m(r_0, A)$, where r_0 is the initial residual. It is therefore mathematically equivalent to FOM. However, because A is symmetric, some simplifications resulting from the three-term Lanczos recurrence leads to a more elegant algorithm. □

3.5 The Singular Value Decomposition

The singular value decomposition, which we will abbreviate “SVD,” is not always the most efficient way of analyzing a linear system, but is extremely flexible, and is sometimes used in signal processing (smoothing), sensitivity analysis, statistical analysis, etc., especially if a large amount of information about the numerical properties of the system is desired. The major libraries for programmers (e.g. Lapack) and software systems (e.g. MATLAB, Mathematica) have facilities for computing the SVD. The SVD is often used in the context of a QR factorization, but the component matrices in an SVD are computed with an iterative technique related to techniques for computing eigenvalues and eigenvectors (in Chapter 5 of this book).

The following theorem defines the SVD.

THEOREM 3.26

Let $A \in L(\mathbb{R}^n, \mathbb{R}^m)$ be otherwise arbitrary. Then there are orthogonal matrices U and V and a matrix $\Sigma = [\Sigma_{ij}] \in L(\mathbb{R}^n, \mathbb{R}^m)$ such that $\Sigma_{ij} = 0$ for $i \neq j$, $\Sigma_{i,i} = \sigma_i \geq 0$ for $1 \leq i \leq p = \min\{m, n\}$, and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p$, such that

$$A = U\Sigma V^T.$$

For a proof and further explanation, see G. W. Stewart, *Introduction to Matrix Computations* [85] or G. H. Golub²⁴ and C. F. van Loan, *Matrix Computations* [34].

Note: The SVD for a particular matrix is not necessarily unique.

Note: The SVD is defined similarly for complex matrices $A \in L(\mathbb{C}^n, \mathbb{C}^m)$.

²⁴Gene Golub, a famous numerical analyst, a professor of Computer Science and, for many years, department chairman, at Stanford University, invented the efficient algorithm used today for computing the singular value decomposition.

REMARK 3.53 A simple algorithm to find the singular-value decomposition is: (1) find the nonzero eigenvalues of $A^T A$, i.e., $\lambda_i, i = 1, 2, \dots, r$, (2) find the orthogonal eigenvectors of $A^T A$ and arrange them in $n \times n$ matrix V , (3) form the $m \times n$ matrix Σ with diagonal entries $\sigma_i = \sqrt{\lambda_i}$, (4) let $u_i = \sigma_i^{-1} A v_i, i = 1, 2, \dots, r$ and compute $u_i, i = r + 1, r + 2, \dots, m$ using Gram-Schmidt orthogonalization. However, a well-known efficient method for computing the SVD is the Golub-Reinsch algorithm [86] which employs Householder bidiagonalization and a variant of the QR method. $\square$

Example 3.18

Let $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$. Then

$$U \approx \begin{pmatrix} -0.2298 & 0.8835 & 0.4082 \\ -0.5247 & 0.2408 & -0.8165 \\ -0.8196 & -0.4019 & 0.4082 \end{pmatrix}, \quad \Sigma \approx \begin{pmatrix} 9.5255 & 0 \\ 0 & 0.5143 \\ 0 & 0 \end{pmatrix}, \quad \text{and}$$

$$V \approx \begin{pmatrix} -0.6196 & -0.7849 \\ -0.7849 & 0.6196 \end{pmatrix} \text{ is a singular value decomposition of } A.$$

$\square$

Note: If $A = U\Sigma V^T$ represents a singular value decomposition of A , then, for $\tilde{A} = A^T$, $\tilde{A} = V\Sigma^T U^T$ represents a singular value decomposition for $\tilde{A}$.

Let $U(:, i)$ denote the i -th column of U and $V(:, i)$ denote the i -th column of V . Then $AV(:, i) = \sigma_i U(:, i)$ for $1 \leq i \leq p$, and, if $n > m$, then $AV(:, i) = 0$ for $p + 1 \leq i \leq n$, that is $\{V(:, i)\}_{i=p+1}^n$ form a basis for the null space of A .

DEFINITION 3.43 The vectors $V(:, i), 1 \leq i \leq p$ are called the right singular vectors of A , while the corresponding $U(:, i)$ are called the left singular vectors of A corresponding to the singular values σ_i .

The singular values are like eigenvalues, and the singular vectors are like eigenvectors. In fact, we have

THEOREM 3.27

Suppose $A \in L(\mathbb{R}^n)$ be symmetric and positive definite. Let $\{\lambda_i\}_{i=1}^n$ be the eigenvalues of A , ordered so that $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, and let v_i be the eigenvector corresponding to λ_i . Furthermore, choose the v_i so $\{v_i\}_{i=1}^n$ is an orthonormal set, and form $V = [v_1, \dots, v_n]$ and $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$. Then $A = V\Lambda V^T$ represents a singular value decomposition of A .

This theorem follows directly from the definition of the SVD.

We also have

THEOREM 3.28

Let $A \in L(\mathbb{R}^n)$ be invertible, and let $A = U\Sigma V^T$ represent a singular value decomposition of A . Then the condition number $\kappa_2(A) = \sigma_1/\sigma_n$.

PROOF $\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2$, while $\|A\|_2 = \max_{\|x\|_2=1} \|Ax\|_2$. However,

$$\|Ax\|_2 = \|U\Sigma(V^T x)\|_2 = \|\Sigma w\|_2, \text{ where } w = V^T x,$$

since $\|Uz\|_2 = \|z\|_2$ for every $z \in \mathbb{R}^n$ (because U is orthogonal). Also, since V is orthogonal, $\|w\|_2 = 1$ if $\|x\|_2 = 1$, and for each $x \in \mathbb{R}^n$, there is a corresponding w , $x = Vw$. Thus,

$$\|Ax\|_2 = \max_{\|w\|_2=1} \|\Sigma w\|_2 = \sigma_1.$$

Now, observe that

$$A^{-1} = V\Sigma^{-1}U^T,$$

where $\Sigma^{-1} = \text{diag}(1/\sigma_1, \dots, 1/\sigma_n)$. Since transposes of orthogonal matrices are orthogonal, a similar argument to that for $\|A\|_2$ shows that $\|A^{-1}\|_2 = 1/\sigma_n$. Therefore,

$$\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2 = \sigma_1(1/\sigma_n).$$

□

Thus, the condition number of a matrix is obtainable directly from the SVD, but the SVD gives us more useful information about the sensitivity of solutions than just that single number, as we'll see shortly.

The singular value decomposition is related directly to the *Moore–Penrose pseudo-inverse*. In fact, the pseudo-inverse can be defined directly in terms of the singular value decomposition.

DEFINITION 3.44 Let $A \in L(\mathbb{R}^n, \mathbb{R}^m)$, let $A = U\Sigma V^T$ represent a singular value decomposition of A , and assume $r \leq p$ is such that $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$, and $\sigma_{r+1} = \sigma_{r+2} = \dots = \sigma_p = 0$. Then the Moore–Penrose pseudo-inverse of A is defined to be

$$A^+ = V\Sigma^+U^T,$$

where $\Sigma^+ = (\Sigma_{ij}^+) \in L(\mathbb{R}^m, \mathbb{R}^n)$ is such that

$$\begin{cases} \Sigma_{ij}^+ = 0 & \text{if } i \neq j \text{ or } i > r, \text{ and} \\ \Sigma_{ii}^+ = 1/\sigma_i & \text{if } 1 \leq i \leq r. \end{cases}$$

Part of the power of the singular value decomposition comes from the following.

THEOREM 3.29

Suppose $A \in L(\mathbb{R}^n, \mathbb{R}^m)$ and we wish to find approximate solutions to $Ax = b$, where $b \in \mathbb{R}^m$. Then,

- If $Ax = b$ is inconsistent, then $x = A^+b$ represents the least squares solution of minimum 2-norm.
- If A is consistent (but possibly underdetermined) then $x = A^+b$ represents the solution of minimum 2-norm.
- In general, $x = A^+b$ represents the least squares solution to $Ax = b$ of minimum norm.

The proof of Theorem 3.29 is left as an exercise (on page 188).

REMARK 3.54 If $m < n$, one would expect the system to be underdetermined but full rank. In that case, A^+b gives the solution x such that $\|x\|_2$ is minimum; however, if A were also inconsistent, then there would be many least squares solutions, and A^+b would be the least squares solution of minimum norm. Similarly, if $m > n$, one would expect there to be a single least squares solution; however, if the rank of A is $r < p = n$, then there would be many such least squares solutions, and A^+b would be the least squares solution of minimum norm. $\square$

Example 3.19

Consider $Ax = b$, where $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ and $b = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$. Then

$$U \approx \begin{pmatrix} -0.2148 & 0.8872 & 0.4082 \\ -0.5206 & 0.2496 & -0.8165 \\ -0.8263 & -0.3879 & 0.4082 \end{pmatrix}, \quad \Sigma \approx \begin{pmatrix} 16.8481 & 0 & 0 \\ 0 & 1.0684 & 0 \\ 0 & 0 & 0.0000 \end{pmatrix},$$

$$V \approx \begin{pmatrix} -0.4797 & -0.7767 & -0.4082 \\ -0.5724 & -0.0757 & 0.8165 \\ -0.6651 & 0.6253 & -0.4082 \end{pmatrix}, \quad \text{and } \Sigma^+ \approx \begin{pmatrix} 0.0594 & 0 & 0 \\ 0 & 0.9360 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Since $\sigma_3 = 0$, we note that the system is not of full rank, so it could be either inconsistent or underdetermined. We compute $x \approx [0.9444, 0.1111, -0.7222]^T$, and we obtain²⁵ $\|Ax - b\|_2 \approx 2.5 \times 10^{-15}$. Thus, $Ax = b$, although apparently

²⁵The computations in this example were done using MATLAB, and were thus done in IEEE double precision. The digits displayed here are the results from that computation, rounded to four significant decimal digits with MATLAB's intrinsic display routines.

underdetermined, is apparently consistent, and x represents that solution of $Ax = b$ which has minimum 2-norm. $\square$

As with other methods for computing solutions, we usually do not form the pseudo-inverse A^+ to compute A^+x , but we use the following.

ALGORITHM 3.9

(Computing A^+b)

INPUT:

- (a) the m by n matrix $A \in L(\mathbb{R}^n, \mathbb{R}^m)$,
- (b) the right-hand-side vector $b \in \mathbb{R}^m$,
- (c) a tolerance ϵ such that a singular value σ_i is considered to be equal to 0 if $\sigma_i/\sigma_1 < \epsilon$.

OUTPUT: an approximation x to A^+b .

1. Compute the SVD of A , that is, compute approximations to $U \in L(\mathbb{R}^m)$, $\Sigma \in L(\mathbb{R}^n, \mathbb{R}^m)$, and $V \in L(\mathbb{R}^n)$ such that $A = U\Sigma V^T$.
2. $p \leftarrow \min\{m, n\}$.
3. $r \leftarrow p$.
4. FOR $i = 1$ to p .
 - IF $\sigma_i/\sigma_1 > \epsilon$ THEN
 - $\sigma_i^+ \leftarrow 1/\sigma_i$.
 - ELSE
 - i. $r \leftarrow i - 1$.
 - ii. EXIT FOR
 - END IF
- END FOR
5. Compute $w = (w_1, \dots, w_r)^T \in \mathbb{R}^r$, $w \leftarrow U(:, 1 : r)^T b$, where $U(:, 1 : r) \in L(\mathbb{R}^r, \mathbb{R}^n)$ is the matrix whose columns are the first r columns of U .
6. FOR $i = 1$ to r : $w_i \leftarrow \sigma_i^+ w_i$.
7. $x \leftarrow \sum_{i=1}^r w_i V(:, i)$.

END ALGORITHM 3.9.

REMARK 3.55 Ill-conditioning (i.e., sensitivity to roundoff error) in the computations in Algorithm 3.9 occurs when small singular values σ_i are used.

For example, suppose $\sigma_i/\sigma_1 \approx 10^{-6}$, and there is an error $\delta U(:, i)$ in the vector b , that is, $b = \tilde{b} - \delta U(:, i)$ (that is, we perturb b by δ in the direction of $U(:, i)$). Then, instead of A^+b ,

$$A^+(b + \delta U(:, i)) = A^+b + A^+\delta U(:, i) = A^+b + \delta \frac{1}{\sigma_i} V(:, i). \quad (3.96)$$

Thus, the norm of the error $\delta U(:, i)$ is magnified by $1/\sigma_i$. Now, if, in addition, b happened to be in the direction of $U(:, 1)$, that is, $b = \delta_1 U(:, 1)$, then $\|A^+b\|_2 = \|\delta_1(1/\sigma_1)V(:, 1)\|_2 = (1/\sigma_1)\|b\|_2$. Thus, the relative error, in this case, would be magnified by σ_1/σ_i . $\square$

In view of Remark 3.55, we are led to consider modifying the problem slightly to reduce the sensitivity to roundoff error. For example, suppose that we are data fitting, with m data points (t_i, y_i) (as in Section 3.3.8 on page 139), and A is the matrix as in Equation (3.26), where $m \gg n$. Then we assume there is some error in the right-hand-side vector b . However, since $\{U(:, i)\}$ forms an orthonormal basis for $\mathbb{R}^m$,

$$b = \sum_{i=1}^m \beta_i U(:, i) \text{ for some coefficients } \{\beta_i\}_{i=1}^m.$$

Therefore, $U^T b = (\beta_1, \dots, \beta_m)^T$, and we see that x will be more sensitive to changes in components of b in the direction of the β_i with larger indices. If we know that typical errors in the data are on the order of $\bar{\epsilon}$, then, intuitively, it makes sense not to use components of b in which the magnification of errors will be larger than that. That is, it makes sense in such cases to choose $\epsilon = \bar{\epsilon}$ in Algorithm 3.9.

Use of $\epsilon \neq 0$ in Algorithm 3.9 can be viewed as replacing the smallest singular values of the matrix A by 0. In the case that $A \in L(\mathbb{R}^n)$ is square and only σ_n is replaced by zero, this amounts to replacing an ill-conditioned matrix A by a matrix that is exactly singular. One (of many possible) theorems dealing with this replacement process is

THEOREM 3.30

Suppose $A \in L(\mathbb{R}^n)$, and we replace $\sigma_n \neq 0$ by 0, then form $\tilde{A} = U\tilde{\Sigma}V^T$, where $A = U\Sigma V^T$ represents the singular value decomposition of A , and $\tilde{\Sigma} = \text{diag}(\sigma_1, \dots, \sigma_{n-1}, 0)$. Then

$$\|A - \tilde{A}\|_2 = \min_{\substack{B \in L(\mathbb{R}^n) \\ \text{rank}(B) < n}} \|A - B\|_2,$$

(The proof is left as Exercise 59 on page 189.)

Suppose now that $\tilde{A}$ has been obtained from A by replacing the smallest singular values of A by 0, so the nonzero singular values of $\tilde{A}$ are $\sigma_1 \geq \sigma_2 \geq$

$\cdots \geq \sigma_r > 0$, and define $x = \tilde{A}^+b$. Then, perturbations of size $\|\Delta b\|$ in b result in perturbations of size at most $(\sigma_1/\sigma_r)\|\Delta b\|$ in x . This prompts us to define a generalization of condition number as follows.

DEFINITION 3.45 Let $A \in L(\mathbb{R}^m, \mathbb{R}^n)$, with m and n arbitrary, and assume the nonzero singular values of A are $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. Then the generalized condition number of A is σ_1/σ_r .

3.6 Exercises

1. Show that the inverse of a positive definite matrix is positive definite.
2. Prove that the diagonal elements of a symmetric positive definite $n \times n$ matrix are positive.
3. A matrix $A = (a_{ij})$ of size $n \times n$ is said to be skew-symmetric if $A^T = -A$. Prove the following properties of a skew-symmetric matrix.
 - (a) $a_{ii} = 0$ for $i = 1, \dots, n$.
 - (b) $I - A$ is nonsingular, where I is the $n \times n$ identity matrix.
4. Define an $n \times n$ real matrix A by $a_{ij} = w_j^T w_i$ for n real vectors $w_1, w_2, \dots, w_n$. Prove that matrix A is real, symmetric, and positive semi-definite.
5. Show that (a), (b), and (c) in Definition 3.19 are satisfied for the second inner product in Example 3.5 on page 92.
6. Prove that, if A is a normal matrix, i.e., $A^H A = A A^H$ then $\|A\|_2 = \rho(A)$. (Note that if A is real symmetric or Hermitian, then A is normal.)
7. Let A be a nonsingular $n \times n$ real matrix and $\|A^{-1}B\| = r < 1$, where the matrix norm is induced from some vector norm.
 - (a) Show that $A + B$ is nonsingular and $\|(A + B)^{-1}\| \leq \frac{\|A^{-1}\|}{1 - r}$.
 - (b) Show that $\|(A + B)^{-1} - A^{-1}\| \leq \frac{\|B\| \|A^{-1}\|^2}{1 - r}$.
8. Find nonzero constants:
 - (a) d_1 and d_2 such that $d_1\|x\|_1 \leq \|x\|_2 \leq d_2\|x\|_1$ for all $x \in \mathbb{R}^n$. Find d_1 and d_2 so each inequality is sharp for some x .

- (b) b_1 and b_2 such that $b_1\|x\|_1 \leq \|x\|_\infty \leq b_2\|x\|_1$ for all $x \in \mathbb{R}^n$. Find b_1 and b_2 so each inequality is sharp for some x .
- (c) Compare with Proposition 3.2.
9. Let $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$ be two vector norms for $\mathbb{R}^n$. Suppose that $\lim_{k \rightarrow \infty} \|z - x_k\|_\beta = 0$. Prove that given $\epsilon > 0$, there is an N such that $\|z - x_k\|_\alpha < \epsilon$ when $k > N$.
10. For a $n \times n$ matrix A , show that, $\|A\|_1 \leq n^{1/2}\|A\|_2 \leq n\|A\|_\infty$.
11. Let

$$A = \begin{pmatrix} 5 & -2 \\ -4 & 7 \end{pmatrix}.$$

- Find $\|A\|_1$, $\|A\|_\infty$, $\|A\|_2$, and $\rho(A)$. Verify that $\rho(A) \leq \|A\|_1$, $\rho(A) \leq \|A\|_\infty$ and $\rho(A) \leq \|A\|_2$.
12. Suppose that matrix A is nonsingular, x is the solution to $Ax = b$, suppose $\|A^{-1}\|_2 = 10^3$, and suppose $\|A\|_2 = 10^2$. We wish to solve $Bz = b$ where $B = A - C$ and $\|C\|_2 = 10^{-4}$.
- (a) Prove that B is nonsingular.
- (b) Find an upper bound on $\|x - z\|_2$ in terms of $\|x\|_2$, that is, find a constant $c > 0$ such that $\|x - z\|_2 < c\|x\|_2$.
13. Let A be $n \times n$ tridiagonal matrix, with

$$a_{ij} = \begin{cases} 5 & \text{if } i = j, \\ 1 & \text{if } i = j + 1 \text{ or } i = j - 1, \\ 0 & \text{otherwise.} \end{cases}$$

- (a) Show that A is nonsingular.
- (b) Show that $\frac{1}{7} \leq \|A^{-1}\|_\infty \leq \frac{1}{3}$.
14. Suppose $\|I - AB_0\| = c < 1$ and

$$B_k = B_{k-1} + B_{k-1}(I - AB_{k-1}),$$

$$k = 1, 2, \dots$$

- (a) Show that $\|I - AB_k\| \leq c^{2^k}$.
- (b) Show that $\|A^{-1} - B_k\| \leq \|B_0\| \frac{c^{2^k}}{1 - c}$.
15. Prove that if λ is an eigenvalue of A and c is a constant, then $1 - \lambda c$ is an eigenvalue of $I - cA$.

16. Using Exercise 15, prove that if $\rho(A) < 1$, then $I - A$ is nonsingular. Also show that $(I - A)^{-1}(I - A^{N+1}) = \sum_{i=0}^N A^i = I + A + A^2 + \cdots + A^N$.
17. Show that back solving for Gaussian elimination (that is, show that completion of Algorithm 3.2) requires $(n^2 + n)/2$ multiplications and divisions and $(n^2 - n)/2$ additions and subtractions.
18. Show that performing the forward phase of Gaussian elimination for $Ax = b$ (that is, completing Algorithm 3.1) requires $\frac{1}{3}n^3 + \mathcal{O}(n^2)$ multiplications and divisions.
19. Show that the inverse of a nonsingular lower triangular matrix is lower triangular.
20. Prove Equation (3.12) on page 109.
21. Explain why $A^{(r+1)} = M^{(r)}A^{(r)}$ and $b^{(r+1)} = M^{(r)}b^{(r)}$, where $A^{(r)}$ is as in Equation (3.10) on page 107 and $M^{(r)}$ is as in Equation (3.11) on page 109.
22. Explain why $A = LU$, where L and U are as in Equation (3.14) on page 109.
23. Show that computing the inverse of a matrix $A \in L(\mathbb{R}^n)$ as in the note on page 110 requires $n^3 + \mathcal{O}(n^2)$ multiplications and divisions.
Hint: To achieve $n^3 + \mathcal{O}(n^2)$ multiplications and divisions, you need to take advantage of the fact that the right-hand sides of the systems you are trying to solve are the unit vectors e_j ; otherwise, the number of multiplications and divisions is $(4/3)n^3 + \mathcal{O}(n^2)$.
24. Prove Proposition 3.10 on page 121. (The exact operation count you get may vary, because certain details of your algorithm, such as taking account of zeros in computing the inverses of the m by m blocks, may vary.)
25. Show that, if U is unitary (that is, if $U^H U = I$), then $\|Ux\|_2 = \|x\|_2$ for every x .
26. Show that, if U is unitary, then $\kappa_2(U) = 1$.
Hint: $\kappa_2(A)$ is the condition number of the matrix A with respect to the 2-norm; see (3.21) on page 122. You may wish to use the result of Exercise 25.
27. Let $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{pmatrix}$ and $b = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$.
- (a) Compute $\kappa_\infty(A)$ approximately.

- (b) Use floating point arithmetic with $\beta = 10$ and $t = 3$ (3-digit decimal arithmetic), rounding-to-nearest, and Algorithms 3.1 and 3.2 to find an approximation to the solution x to $Ax = b$.
 - (c) Execute Algorithm 3.5 by hand, using $t = 3$, $\beta = 10$, and outwardly rounded interval arithmetic (and rounding-to-nearest for computing Y).
 - (d) Find the exact solution to $Ax = b$ by hand.
 - (e) Compare the results you have obtained.
28. Derive the normal equations (3.29) from (3.28).

29. Let $A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 4 \\ 5 \\ 8 \end{pmatrix}$.

- (a) Compute a QR factorization of A using Householder transformations. (You may use a tool such as MATLAB's, but show each step.)
 - (b) Compute a QR factorization of A using Givens rotations, showing each step.
 - (c) Compute a QR factorization using some preprogrammed routine (such as MATLAB's "**qr**" function). Are the three QR factorizations you have obtained the same?
 - (d) Use one of the QR factorizations to compute the least squares solution to $Ax = b$.
 - (e) If the three QR factorizations are not the same, then compute the least squares solution to $Ax = b$ using each of the distinct factorizations. Are the answers that you get the same, to within roundoff error?
30. Let

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 4 & 4 & 1 \\ 6 & -5 & 8 \end{pmatrix}.$$

- (a) Find the LU factorization of A , such that L is lower triangular and U is unit upper triangular.
 - (b) Perform back solving then forward solving to find a solution x for the system of equations $Ax = b = [4 \ 7 \ 15]^T$.
31. Let $A \in \mathbb{R}^{(p+q) \times (p+q)}$ be a nonsingular matrix given by

$$A = \begin{pmatrix} C & B^T \\ B & 0 \end{pmatrix},$$

where $C \in \mathbb{R}^{p \times p}$ and $B \in \mathbb{R}^{q \times p}$. Assume A has an LU decomposition with

$$L = \begin{pmatrix} I & 0 \\ X & I \end{pmatrix} \text{ and } U = \begin{pmatrix} C & Y \\ 0 & Z \end{pmatrix}.$$

Find X , Y , and Z .

32. Let the $n \times n$ matrix A have elements

$$a_{ij} = \int_0^1 e^{ix} e^{jx} dx$$

for $1 \leq i, j \leq n$. Prove that A has a Cholesky factorization $A = L^T L$.

33. Find the Cholesky factorization of

$$A = \begin{pmatrix} 1 & -1 & 2 \\ -1 & 5 & 4 \\ 2 & 4 & 29 \end{pmatrix}.$$

Also explain why A is positive definite.

34. Let A be an invertible matrix. Suppose $A, \Delta A \in \mathbb{R}^{n \times n}$ and $b, \Delta b \in \mathbb{R}^n$ are such that $Ax = b$ and $(A + \Delta A)y = b + \Delta b$. Further assume that

- (i) $\|\Delta A\| \leq \delta \|A\|$,
- (ii) $\|\Delta b\| \leq \delta \|b\|$, and
- (iii) $\delta \kappa(A) = r < 1$, where $\kappa(A)$ is the condition number of A .

Then

- (a) Show that $A + \Delta A$ is nonsingular.

- (b) Prove that $\frac{\|y\|}{\|x\|} \leq \frac{1+r}{1-r}$.

35. Let $A = \begin{pmatrix} 0.1\alpha & 0.1\alpha \\ 1.0 & 1.5 \end{pmatrix}$. Determine α such that $\kappa(A)$ is minimized. Use the maximum norm.

36. Let A be $n \times n$ lower triangular matrix with elements

$$a_{ij} = \begin{cases} 1 & \text{if } i = j, \\ -1 & \text{if } i = j + 1, \\ 0 & \text{otherwise.} \end{cases}$$

Determine the condition number of A using the matrix norm $\|\cdot\|_\infty$.

37. Let $A = I - x x^T$ where $x \in \mathbb{R}^n$ and $\|x\|_2 = \sqrt{2}$. Prove that condition number $\kappa_2(A) = \|A^{-1}\|_2 \|A\|_2 = 1$ (i.e. A is *perfectly conditioned*).

38. Consider solving $Ax = b$ by decomposing,

$$A = \begin{pmatrix} a_1 & c_1 & 0 \\ b_2 & a_2 & c_2 \\ 0 & b_3 & a_3 \end{pmatrix} = \begin{pmatrix} \alpha_1 & 0 & 0 \\ b_2 & \alpha_2 & 0 \\ 0 & b_3 & \alpha_3 \end{pmatrix} \begin{pmatrix} 1 & \gamma_1 & 0 \\ 0 & 1 & \gamma_2 \\ 0 & 0 & 1 \end{pmatrix}.$$

Let $|a_1| > |c_1| > 0$, $|a_2| \geq |b_2| + |c_2|$ and $|a_3| > |b_3|$. Show that

(a) $\alpha_i \neq 0$ for $i = 1, 2, 3$.

(b) $|\gamma_i| < 1$ for $i = 1, 2$.

39. Consider the matrix system $Au = b$ given by

$$\begin{pmatrix} \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & 0 & 0 \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & 0 \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{4} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

(a) Determine A^{-1} by hand.

(b) Determine the infinity-norm condition number of the matrix A .

(c) Let $\tilde{u}$ be the solution when the right-hand side vector b is perturbed to $\tilde{b} = (1.01 \ 0 \ 0 \ 0.99)^T$. Estimate $\|u - \tilde{u}\|_\infty$, without computing $\tilde{u}$.

40. Let $x = \begin{pmatrix} 3 \\ 0 \\ 4 \end{pmatrix}$. Find a Householder matrix H such that $Hx = \begin{pmatrix} -5 \\ 0 \\ 0 \end{pmatrix}$.

41. Consider solving the matrix system $Ax = b$ by first factoring A into LU via the *Doolittle Algorithm*, then solving $Ly = b$ and $Ux = y$. The Doolittle LU algorithm is given by:

input $n, (a_{ij})$

for $k = 1$ to n do

$l_{kk} = 1$

for $j = k$ to n do

$$u_{kj} = a_{kj} - \sum_{s=1}^{k-1} l_{ks} u_{sj}$$

end do

for $i = k + 1$ to n do

$$l_{ik} = \frac{\left(a_{ik} - \sum_{s=1}^{k-1} l_{is} u_{sk} \right)}{u_{kk}}$$

end do

end do

output $(l_{ij}), (u_{ij})$

(a) Show that LU factorization algorithm given above requires

$$\frac{n^3}{3} - \frac{n}{3}$$

multiplications and divisions and

$$\frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6}$$

additions and subtractions.

- (b) Show that solving $Ly = b$, where L is lower-triangular with $l_{ii} = 1$ for all i , requires $n^2/2 - n/2$ multiplications and divisions and $n^2/2 - n/2$ additions and subtractions.
- (c) Show that solving $Ux = y$, where U is upper-triangular requires $n^2/2 + n/2$ multiplications and divisions and $n^2/2 - n/2$ additions and subtractions.
- (d) Add all operation counts in parts (a), (b), (c) and compare with Gaussian Elimination.

42. For the system as in Example 3.14 (on page 153),

- (a) show that $\rho(J) < 1$, but $\rho(\mathcal{G}) = 1$;
- (b) try $x^{(0)} = (1, 1, 1)^T$ in the Gauss–Seidel method, and see what happens.

43. Complete the computations, to check that $x^{(4)}$ is as given in Example 3.15 on page 155.

44. Use Lemma 3.11 to verify the statements in Example 3.16 (on page 157).

45. Verify Equation (3.88) on page 169.

46. Let A be the point matrix in Example 3.9 (on page 129), and let $\mathbf{b}$ be the point right-hand-side vector. Apply the interval Gauss–Seidel method (Equation (3.58)) to $Ax = b$, starting with $\mathbf{x}_i^{(0)} = [-10, 10]$, $1 \leq i \leq 3$, using the inverse midpoint preconditioner, using double

precision IEEE arithmetic and outward rounding,²⁶ and iterating until the result is apparently stationary. Have you proven that the system in Example 3.9 has a solution? If so, then what rigorous bounds on that solution do you get?

47. Repeat Exercise 46, but with interval Gaussian elimination (as in §3.3.7 starting on page 130) instead of the interval Gauss–Seidel method. Compare the results.
48. Let A be the $n \times n$ tridiagonal matrix with,

$$a_{ij} = \begin{cases} 4 & \text{if } i = j, \\ -1 & \text{if } i = j + 1 \text{ or } i = j - 1, \\ 0 & \text{otherwise.} \end{cases}$$

Prove that the Gauss-Seidel and Jacobi methods converge for this matrix.

49. Prove that if the matrix $A = M - N$ is singular and M is nonsingular, then $\|M^{-1}N\| \geq 1$, where $\|\cdot\|$ is any induced matrix norm.
50. Prove that if matrix $A = \begin{pmatrix} \alpha & \beta \\ \beta & \gamma \end{pmatrix}$ is positive definite, then the Jacobi method converges for a linear system $Ax = b$. (You prove that the Jacobi method converges for 2×2 positive definite matrices. This does not contradict Example 3.12 or the note preceding this example.)
51. Let $A = D - U - L$, where A is strictly diagonally dominant, D is diagonal, U is upper triangular and L is lower triangular. Furthermore, assume that D , U , and L have all nonnegative elements, that is, $D, U, L \geq 0$. Suppose $b \geq 0$. Consider the Gauss-Seidel iterative procedure

$$x^{(m+1)} = (D - L)^{-1}Ux^{(m)} + (D - L)^{-1}b$$

for $m = 0, 1, 2, \dots$, with $x^{(0)} = 0$. Assume also that the spectral radius obeys

$$\rho((D - L)^{-1}U) = \gamma < 1.$$

Prove that $x^{(m)} \rightarrow x$, where all the elements of x are nonnegative, that is, $x \geq 0$. (Hint: Show that $x^{(m)} \geq 0$ for each m .)

52. Consider the linear system $Ax = b$, where $A = L + D + U$, L is strictly lower triangular, D is diagonal, and U is strictly upper triangular. The SOR iterative method has the form

$$x^{(k+1)} = T_\sigma x^{(k)} + c,$$

²⁶such as in INTLAB

where

$$c = \left(L + \frac{1}{\sigma} D \right)^{-1} b$$

and

$$T_\sigma = (\sigma L + D)^{-1}[(1 - \sigma)D - \sigma U].$$

Let

$$A = \begin{pmatrix} 2 & -5 \\ 1 & 2 \end{pmatrix} \quad \text{and} \quad x^{(0)} = b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Prove that the SOR method with $\sigma = 1$ is not convergent, but the SOR method with $\sigma = \frac{1}{2}$ is convergent.

53. Suppose that A is irreducibly diagonally dominant and $A = L + D + U$ with $D = aI$. Let λ be an eigenvalue of A . Prove that $|\lambda/a - 1| < 1$.
54. Consider the linear system,

$$\begin{pmatrix} 3 & 2 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 7 \\ 10 \end{pmatrix}.$$

Using the starting vector $x^{(0)} = (0, 0)^T$, carry out two iterations of the Conjugate Gradient Method to solve the system.

55. Consider the two iterative methods considered in section 3.4.11. Let

$$A = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix} \quad \text{and} \quad F = \begin{pmatrix} 4.75 & -1.875 \\ -1.875 & 0.90 \end{pmatrix}.$$

- (a) Find $\|I - AF\|_\infty$.
- (b) Next, find t that will guarantee that $\|x_t - A^{-1}\|_\infty \leq 10^{-8}$ for each of the iteration methods described in section 3.4.11.
56. Suppose that y_1 and y_2 are linearly independent eigenvectors of $n \times n$ matrix A with eigenvalues λ_1 and λ_2 with $\lambda_1 \neq \lambda_2$. Suppose that $r_0 = b - Ax_0$ satisfies $r_0 = c_1 y_1 + c_2 y_2$. Let $q_1 = \frac{r_0}{\|r_0\|}$ in Arnoldi's method.
- (a) Show that the Arnoldi algorithm stops at $k = 2$.
- (b) Let $y_2 = y_0 + w$ where w belongs to the Krylov subspace $K_2(A, r_0)$. Show that $Ax_2 = b$. (Note that $b - Ax_2 \perp K_2(A, r_0)$.)
57. Prove Theorem 3.29 on page 177. (Hint: You may need to consider various cases. In any case, you'll probably want to use the properties of orthogonal matrices, as in the proof of Theorem 3.28.)
58. Given U , Σ , and V as given in Example 3.19, compute A^+b by using Algorithm 3.9. How does the x that you obtain compare with the x reported in Example 3.19?

59. Prove Theorem 3.30 (on page 179).

60. Suppose $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{pmatrix}$ can be written $A = U\Sigma V^T$, where U and V are orthogonal,

$$U \approx \begin{pmatrix} 0.2093 & 0.9644 & 0.1617 \\ 0.5038 & 0.0353 & -0.8631 \\ 0.8380 & -0.2621 & 0.4785 \end{pmatrix},$$

$$\Sigma \approx \begin{pmatrix} 17.4125 & 0 & 0 \\ 0 & 0.8752 & 0 \\ 0 & 0 & 0.1969 \end{pmatrix}, \text{ and}$$

$$V \approx \begin{pmatrix} 0.4647 & -0.8333 & 0.2995 \\ 0.5538 & 0.0095 & -0.8326 \\ 0.6910 & 0.5528 & 0.4659 \end{pmatrix}.$$

Suppose we want to solve the system $Ax = b$, where $b = [1, -1, 1]^T$, but that, due to noise in the data, we do not wish to deal with any system of equations with condition number equal to 25 or greater. Use the above singular value decomposition to write down the solution of minimum norm to a rank-two system of equations nearest to $Ax = b$, such that the computations proceed with a matrix whose condition number is less than 25. Explain what we mean by “nearest” here.

61. Find the singular value decomposition of the matrix $A = \begin{pmatrix} 1 & 2 \\ 1 & 1 \\ 1 & 3 \end{pmatrix}$.

62. Consider the singular value decomposition of the matrix $A \in \mathbb{R}^{m \times n}$ given by $A = PDQ$, where P is an $m \times m$ unitary matrix, D is an $m \times n$ diagonal matrix, and Q is an $n \times n$ unitary matrix. Show that $\|A(A^T A)^{-1} A^T\|_2 = 1$, assuming that $A^T A$ is nonsingular. (Note that D is not a square matrix.)

Chapter 4

Approximation Theory

4.1 Introduction

In this chapter, we consider approximation of, for example, $f \in C[a, b]$, where $C[a, b]$ represents the space of continuous functions on the interval $[a, b]$. We are interested in approximating $f(x)$ by an elementary function $p(x)$ on the interval $[a, b]$. For example, $p(x)$ could be a polynomial of degree n , a continuous piecewise linear function, a trigonometric polynomial, a rational function, or a linear combination of “nice” functions, that is, functions that are easy to use in numerical computation.

We study approximation of functions in the setting of normed vector spaces. We begin with a review of normed linear spaces, inner products, projections, and orthogonalization.

4.2 Norms, Projections, Inner Product Spaces, and Orthogonalization in Function Spaces

Recall the basic properties of normed vector spaces, which we review in Section 3.2 on page 88. Here, we will be using norms in the context of vector spaces without a finite-dimensional basis.

Example 4.1

Here, V is our vector space.

(a) $V = C[a, b]$. Two common norms are:

$$(a1) \|v\|_{\infty} = \max_{a \leq x \leq b} (|v(x)|\rho(x)),$$

where $\rho(x) > 0$ on $[a, b]$ and $\rho \in C[a, b]$. This is called the *Chebyshev*, *uniform*, or *max norm* with weight function $\rho(x)$.

$$(a2) \|v\|_2 = \left(\int_a^b |v(x)|^2 \rho(x) dx \right)^{\frac{1}{2}},$$

where $\rho(x) > 0$, $a < x < b$, $\rho \in C(a, b)$, and

$$\int_a^b \rho(x) dx < \infty.$$

This is called the L^2 -norm or *least squares norm* with weight function $\rho(x)$.

- (b) If $V = \mathbb{R}^n$, then $\|\cdot\|_1$, $\|\cdot\|_2$, and $\|\cdot\|_\infty$ are norms for $\mathbb{R}^n$. (See Section 3.2 on page 88 and the subsequent pages.)

□

4.2.1 Best Approximations

Now let W be a finite-dimensional subspace of V . A typical problem in approximation theory is: Given $v \in V$, find $w \in W$ such that the distance $\|v - w\|$ is least among all $w \in W$. Such a w is called a *best approximation in W to v* with respect to norm $\|\cdot\|$. (For example, $V = C[a, b]$, $\|\cdot\| = \|\cdot\|_\infty$, and $W = \{\text{set of polynomials of degree } \leq n\}$.)

Question: Does such a w exist? We will prove that the answer to this is yes.

THEOREM 4.1

(Existence of a best approximation) *Let W be an $n + 1$ -dimensional subspace of a normed linear space V . Let $u_0, u_1, \dots, u_n$ be linearly independent elements of W . (Thus, $W = \text{span}(u_0, u_1, u_2, \dots, u_n)$.) Then there is a $p \in W$, i.e., $p = \sum_{j=0}^n \alpha_j u_j$ for a given $f \in V$, such that*

$$\|f - p\| = \left\| f - \sum_{j=0}^n \alpha_j u_j \right\| = \min_{\gamma_0, \gamma_1, \dots, \gamma_n} \left\| f - \sum_{j=0}^n \gamma_j u_j \right\|,$$

that is, $\|f - p\| \leq \|f - q\|$ for all $q \in W$. (p is the best approximation to $f \in V$ with respect to norm $\|\cdot\|$.) This is illustrated schematically in Figure 4.1.

PROOF The proof is divided into two cases.

Case 1: Suppose that f is linearly dependent on $u_0, u_1, \dots, u_n$. Then

$$f = \sum_{j=0}^n \alpha_j u_j = p,$$

and $\|f - p\| = 0$. That is, $f \in W$, so $p = f$.

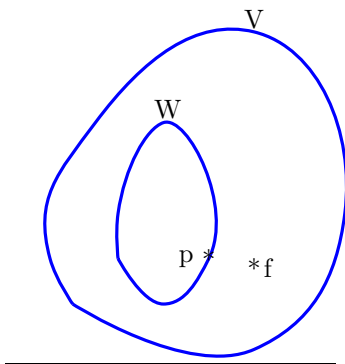


FIGURE 4.1: p is the best approximation to $f \in V$.

Case 2: Suppose that $f \notin W$, so f is linearly independent of $u_0, u_1, \dots, u_n$. Let

$$E = \left\{ w \in V : w = \sum_{j=0}^n z_j u_j + z_{n+1} f \right\},$$

where $z_j \in \mathbb{R}$ for $j = 0, 1, \dots, n+1$. Then E is an $(n+2)$ -dimensional subspace of V . Let $z = (z_0, z_1, \dots, z_{n+1})^T \in \mathbb{R}^{n+2}$, where $z_0, z_1, \dots, z_{n+1}$ are given. Let $\|\cdot\|_*$ be defined on $\mathbb{R}^{n+2}$ by

$$\|z\|_* = \left\| \sum_{j=0}^n z_j u_j + z_{n+1} f \right\|.$$

Then,

1. $\|z\|_* \geq 0$ and $\|z\|_* = 0$ if and only if $z = 0$, since $u_0, u_1, \dots, u_n, f$ are independent,
2. $\|\lambda z\|_* = |\lambda| \|z\|_*$, and
3. $\|z_1 + z_2\|_* \leq \|z_1\|_* + \|z_2\|_*$,

because $\|\cdot\|$ has these properties. Thus, $\|\cdot\|_*$ is a norm on $\mathbb{R}^{n+2}$, i.e., $\|\cdot\|_* : \mathbb{R}^{n+2} \rightarrow \mathbb{R}$. Now, define p to be such that

$$\|f - p\| = \left\| f - \sum_{j=0}^n \alpha_j u_j \right\| = \min_{\gamma_0, \gamma_1, \dots, \gamma_n} \left\| f - \sum_{j=0}^n \gamma_j u_j \right\|,$$

assuming that p exists. But

$$\|f - p\| = \left\| f - \sum_{j=0}^n \alpha_j u_j \right\| = \|e - a\|_*,$$

where $e = (0, 0, \dots, 0, 1)^T$ and $a = (\alpha_0, \alpha_1, \dots, \alpha_n, 0)^T$. Clearly,

$$\|e - a\|_* = \min_{z \in G} \|e - z\|_*,$$

where $G = \{z \in \mathbb{R}^{n+2} : z = (z_0, z_1, \dots, z_n, 0)^T\}$. Thus, if $a \in \mathbb{R}^{n+2}$ exists such that $\|e - a\|_* = \min_{z \in G} \|e - z\|_*$, then

$$p = \sum_{j=0}^n \alpha_j u_j \in V$$

exists. (We have reduced the problem of finding $p \in W$ to finding $a \in \mathbb{R}^{n+2}$.)

Question: Can we find a ?

Define $H = \{z \in G : \|z - e\|_* \leq \|e\|_*\}$. Then,

(a) H is not empty, since $0 = (0, 0, \dots, 0)^T \in H$.

(b) H is bounded in $\mathbb{R}^{n+2}$ since if $z \in H$, then

$$\|z\|_* \leq \|z - e\|_* + \|e\|_* \leq 2\|e\|_*.$$

Therefore, z has an infimum¹ $\mu = \inf_{z \in H} \|e - z\|_*$. Let

$$z^{(t)} = (z_0^{(t)}, z_1^{(t)}, \dots, z_n^{(t)}, 0) \in H \subset \mathbb{R}^{n+2}$$

be a sequence of vectors such that $\|e - z^{(t)}\|_* \rightarrow \mu$ as $t \rightarrow \infty$. By the Bolzano–Weierstrass Theorem (i.e. every bounded sequence in $\mathbb{R}^m$ has a convergent subsequence), a subsequence of $\{z^{(t)}\}$ has a limit point $\hat{z}$. Thus, $a = \hat{z}$. This, in turns, implies existence of $p \in W$. $\square$

Example 4.2

Find the best approximation $p \in P^0$ to $f(x) = e^x \in C[0, 1]$ for the $\|\cdot\|_\infty$ and $\|\cdot\|_2$ norms. (Thus, $W = P^0 \subset V = C[0, 1]$.)

(a) Find p that minimizes $\|e^x - p\|_\infty = \max_{0 \leq x \leq 1} |e^x - p|$.

In this case, $p = \frac{1}{2}(e + 1)$ and $\max_{0 \leq x \leq 1} |e^x - p| = \frac{1}{2}(e - 1)$.

(b) Find p that minimizes $\|e^x - p\|_2 = \left(\int_0^1 (e^x - p)^2 dx \right)^{\frac{1}{2}}$.

In this case, $p = e - 1$ and $\|e^x - p\|_2 = \frac{1}{2}(4e - e^2 - 3)^{\frac{1}{2}}$.

¹A fundamental property of a finite-dimensional vector space is that every set that is bounded below has a greatest lower bound, or infimum.

□

We will now see that approximation in inner product spaces is straightforward. We introduced the concept of inner product spaces in the context of matrix computations (i.e. finite-dimensional vector spaces) in Definition 3.19 on page 91. We review this same concept here, in the more general context of function spaces.

DEFINITION 4.1 *A real vector space V is a real inner product space if for each $u, v \in V$, a real number (u, v) can be defined with the properties:*

- (i) $(u, u) \geq 0$ for $u \in V$ with $(u, u) = 0$ if and only if $u = 0$,
- (ii) $(u, v) = (v, u)$, and
- (iii) $(\alpha u + \beta v, w) = \alpha(u, w) + \beta(v, w)$ for all $u, v, w \in V$ and $\alpha, \beta \in \mathbb{R}$.

Example 4.3

(of inner product spaces)

$$(1) \quad V = C[a, b] \text{ with } (f, g) = \int_a^b \rho(x) f(x) g(x) dx, \text{ where } \rho(x) > 0 \text{ for } a \leq x \leq b, \text{ and } \rho \in C[a, b].$$

$$(2) \quad \mathbb{R}^n \text{ with } (x, y) = x^T y.$$

□

Unless we specify otherwise, when we say “inner product space” or “normed linear space” in the remainder of this chapter, we will mean “real inner product space.” Much of what is presented is also true in spaces over the complex numbers.

REMARK 4.1 Complex inner product spaces can be defined analogously, with the following modifications to properties (ii) and (iii) of Definition 4.1:

- (ii)' $(u, v) = \overline{(v, u)}$,
- (iii)' $(\alpha u + \beta v, w) = \overline{\alpha}(u, w) + \overline{\beta}(v, w)$ for all $u, v, w \in V$ and $\alpha, \beta \in \mathbb{C}$.

Complex inner product spaces corresponding to Example 4.3 are

$$(1) \quad \mathcal{V} = f : [a, b] \rightarrow \mathbb{C}, \quad f \text{ continuous, with } (f, g) = \int_a^b \rho(x) \overline{f(x)} g(x) dx, \\ \text{where } \rho(x) > 0 \text{ for } a \leq x \leq b, \text{ and } \rho \in C[a, b].$$

(2) $\mathbb{C}^n$ with $(z, w) = z^H w$, where z^H is the conjugate transpose of z .

We will work with complex inner product spaces when we study trigonometric approximation in §4.5. $\square$

THEOREM 4.2

Any real inner product space V is a real normed linear space with norm defined by $\|v\| = (v, v)^{\frac{1}{2}}$.

PROOF Clearly (i), (ii), and (iii) of Definition 3.13 (the definition of norm, on page 88) are satisfied, while (iv) follows from the Cauchy–Schwarz inequality

$$|(u, v)| \leq \|u\| \|v\| \quad \forall u, v \in V.$$

In particular,

$$\begin{aligned} \|u + v\|^2 &= (u + v, u + v) \\ &= (u, u) + 2(u, v) + (v, v) \\ &\leq \|u\|^2 + 2\|u\| \|v\| + \|v\|^2 = (\|u\| + \|v\|)^2. \end{aligned}$$

$\square$

Example 4.4

(norms on inner product spaces)

(1) $V = C[a, b]$ is an inner product space with inner product

$$(f, g) = \int_a^b f(x)g(x)dx$$

and a normed linear space with the norm

$$\|f\| = \left(\int_a^b f^2(x)dx \right)^{\frac{1}{2}}.$$

The Cauchy–Schwarz inequality for this inner product has the form

$$\left| \int_a^b f(x)g(x)dx \right| \leq \left(\int_a^b f^2(x)dx \right)^{\frac{1}{2}} \left(\int_a^b g^2(x)dx \right)^{\frac{1}{2}}.$$

(2) $V = \mathbb{R}^n$ is an inner product space with inner product

$$(x, y) = \sum_{i=1}^n x_i y_i = x^T y$$

and a normed space with norm

$$\|x\| = \left(\sum_{i=1}^n x_i^2 \right)^{\frac{1}{2}}.$$

The Cauchy–Schwarz inequality for this inner product is

$$\left| \sum_{i=1}^n x_i y_i \right| \leq \left(\sum_{i=1}^n x_i^2 \right)^{\frac{1}{2}} \left(\sum_{i=1}^n y_i^2 \right)^{\frac{1}{2}}.$$

□

REMARK 4.2 The concept of a *Cauchy sequence* (Definition 2.3 on page 40) can be generalized to normed linear spaces. A sequence of vectors $u_1, u_2, \dots$ in a normed linear space is said to be a Cauchy sequence if, given any $\epsilon > 0$, there is an integer $N = N(\epsilon)$ such that $\|u_n - u_m\| < \epsilon$ for all $m, n \geq N$. It is easy to show that every convergent sequence is a Cauchy sequence. A Cauchy sequence, however, may not be convergent to an element of the space. If every Cauchy sequence in V is convergent, we say that V is complete.² A complete normed linear space is called a *Banach space*. A complete inner product space is called a *Hilbert space*. □

Example 4.5

(of Hilbert and Banach spaces)

- (1) Let ℓ_2 = set of all real sequences $u = \{u_1, u_2, \dots\} = \{u_i\}_{i=1}^{\infty}$ that satisfy $\sum_{i=1}^n |u_i|^2 < \infty$. Define

$$(u, v) = \sum_{j=1}^{\infty} u_j v_j.$$

Then ℓ_2 is an inner product space, and ℓ_2 can be shown to be complete. Thus, ℓ_2 is a Hilbert space.

- (2) $C[a, b]$ with norm $\|\cdot\|_{\infty}$ is complete and is thus a Banach space.

□

²Basically, this means that the space contains all of its limit points.

4.2.2 Best Approximations in Inner Product Spaces

Consider now the problem of finding the best approximation in an inner product space. Specifically, we wish to find $w \in W \subset V$ that is closest to a given $v \in V$, where V is an inner product space and W is finite-dimensional. Let $W = \text{span}(w_1, w_2, \dots, w_n) \subset V$, where $\{w_i\}_{i=1}^n$ is a linearly independent set. We wish to find $w \in W$ such that $\|w - v\|^2 \leq \|u - v\|^2$ for all $u \in W$. But

$$\begin{aligned} \|w - v\|^2 &= (w - v, w - v) \\ &= \left(\sum_{j=1}^n \alpha_j w_j - v, \sum_{k=1}^n \alpha_k w_k - v \right) \\ &= \sum_j \sum_k \alpha_j \alpha_k (w_j, w_k) - \sum_j \alpha_j (v, w_j) - \sum_k \alpha_k (v, w_k) + (v, v) \\ &= F(\alpha_1, \alpha_2, \dots, \alpha_n), \end{aligned}$$

where $w = \sum_{j=1}^n \alpha_j w_j$.

Thus, the problem reduces to finding the minimum of F as a function of $\alpha_1, \alpha_2, \dots, \alpha_n$. Hence, setting $\partial F / \partial \alpha_\ell = 0$ gives

$$\sum_{j=1}^n \alpha_j (w_j, w_\ell) - (v, w_\ell) = 0,$$

or

$$\sum_{j=1}^n \alpha_j (w_j, w_\ell) = (v, w_\ell) \text{ for } \ell = 1, 2, \dots, n. \quad (4.1)$$

Equations (4.1) represent a linear system that is positive definite and hence invertible, and thus can be solved for the α_j . In this case, the best approximation w is called the least-squares approximation. (The matrix corresponding to the system (4.1) is often called the *Gram matrix*.)

REMARK 4.3 Compare this with our explanation in Section 3.3.8.4 on page 139. In particular, in both (3.29) and (4.1), the system of equations is called the *normal equations*. The functions φ in Section 3.3.8.4 correspond to the vectors w here. The dot product in Section 3.3.8.4 is the finite dot product

$$(\varphi^{(i)}, \varphi^{(j)}) = \sum_{k=1}^m \varphi^{(i)}(t_k) \varphi^{(j)}(t_k).$$

The latter is a dot product only if $\{\varphi^{(i)}\}_{i=1}^m$ is “linearly independent on the finite set $\{t_k\}_{k=1}^m$.” □

The following concept is closely related to finding best approximations in Hilbert spaces.

DEFINITION 4.2 *Let W be a finite-dimensional subspace of an inner product space V . An operator P that maps V into W such that $P^2 = P$ is called a projection operator from V into W .*

REMARK 4.4 Projections are another useful way of defining approximations. For example, $P : V \rightarrow W$ can be defined as

$$Pv = \sum_{k=1}^n \alpha_k w_k,$$

where the α_k 's satisfy

$$\sum_{k=1}^n \alpha_k (w_k, w_\ell) = (v, w_\ell) \text{ for } \ell = 1, 2, \dots, n$$

and $\{w_1, \dots, w_n\}$ is a basis for W . In this example, P is a “least squares” projection operator. $\square$

We now revisit the concept of orthonormal sets of vectors, which we originally introduced on page 92 in conjunction with QR factorizations.

DEFINITION 4.3 *(a restatement of Definition 3.21) Let V be an inner product space. Two vectors u and v in V are called orthogonal if $(u, v) = 0$. A set of such vectors that are pairwise orthogonal is called orthonormal, provided $(u, u) = 1$ for every vector u in that set.*

Let $w_1, w_2, \dots, w_m$ be an orthonormal set in V , i.e., $(w_i, w_j) = \delta_{ij}$. Let

$$M = \text{span}(w_1, w_2, \dots, w_m)$$

be the subspace in V spanned by $w_1, w_2, \dots, w_m$, i.e., given $v \in M$, $v = \sum_{i=1}^m c_i w_i$. Define

$$M^\perp = \{v \in V : (v, w) = 0 \text{ for every } w \in M\},$$

i.e., the elements of $M^\perp$ are orthogonal to those of M .

DEFINITION 4.4 $M^\perp$ is called the orthogonal complement of M in V .

REMARK 4.5 $M^\perp$ is a subspace of V . That is,

- (a) $0 \in M^\perp$,
- (b) if $v_1, v_2 \in M^\perp$ then $v_1 + v_2 \in M^\perp$,
- (c) if $v_1 \in M^\perp$ then $c_1 v_1 \in M^\perp$.

□

REMARK 4.6 Given $u \in V$, we can associate with u two vectors Pu and Qu , with

$$u = Pu + Qu,$$

where

$$Pu = \sum_{k=1}^m (u, w_k) w_k \quad \text{and} \quad Qu = u - \sum_{k=1}^m (u, w_k) w_k.$$

Clearly, $Pu \in M$ and $Qu \in M^\perp$.

□

DEFINITION 4.5 The vector Pu is the orthogonal projection of u onto M and Qu is the perpendicular from u onto M .

We now have

PROPOSITION 4.1

(Projections and best approximation) Let V be an inner product space. Given $u \in V$, $\|u - Pu\| \leq \|u - h\|$ for any $h \in M$, where $\|\cdot\| = (\cdot, \cdot)^{\frac{1}{2}}$. Thus, the vector in M closest to $u \in V$ is Pu , i.e., Pu is the best approximation in M to $u \in V$ with respect to norm $\|\cdot\|$.

PROOF Let $a = h - Pu$, $a \in M$. Then $h = a + Pu$. Thus,

$$\|h - u\|^2 = \|a + Pu - u\|^2 = \|a + c\|^2,$$

where $c = Pu - u$. Continuing,

$$\|h - u\|^2 = (a + c, a + c) = (a, a) + (c, c),$$

since $c = -Qu \in M^\perp$ and $a \in M$, so $(a, c) = 0$. Therefore,

$$\|h - u\|^2 = \|a\|^2 + \|Pu - u\|^2,$$

so $\|Pu - u\| \leq \|h - u\|$ for any $h \in M$.

□

REMARK 4.7 Notice how easy it is to find a best approximation in an inner product space from a finite-dimensional subspace with an orthonormal basis.

□

REMARK 4.8 Consider this proposition geometrically for $V = \mathbb{R}^3$ and

$$M = \text{span}(w_1, w_2) = \text{span}((1, 0, 0)^T, (0, 1, 0)^T).$$

(See Figure 4.2.) Notice that M is the xy -plane. The vector in M closest to u is

$$Pu = \sum_{k=1}^2 (u, w_k) w_k = u_1 w_1 + u_2 w_2,$$

and $Qu = u_3(0, 0, 1)^T$. □

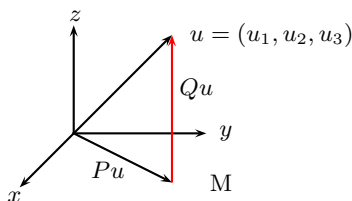


FIGURE 4.2: The projection is the best approximation. See Prop. 4.1.

REMARK 4.9 By Proposition 4.1, $\|u - Pu\| = \|Qu\|$ is the shortest distance from subspace M to u in V . □

4.2.3 The Gram–Schmidt Process (Formal Treatment)

Suppose now M has a basis $\{u_1, u_2, \dots, u_m\}$ that is not orthonormal. Can we find an orthogonal basis $w_1, w_2, \dots, w_n$? (Recall how easy it is to find a best approximation in M to V if M has an orthogonal basis.) This motivates the well-known Gram–Schmidt orthogonalization process.

In Section 3.3.8 (on page 138), we briefly introduced the Gram–Schmidt process in the context of QR -factorizations. Here, we carefully consider the process formally.

THEOREM 4.3

(Gram–Schmidt Process)

Let $u_1, u_2, \dots, u_m$ be linearly independent vectors (elements) in inner product

space V . If

$$\begin{aligned} v_1 &= u_1 \\ v_j &= u_j - \sum_{k=1}^{j-1} \frac{(u_j, v_k)v_k}{(v_k, v_k)} \quad \text{for } j = 2, 3, \dots, m, \text{ and} \\ w_j &= \frac{v_j}{\|v_j\|} \quad \text{for } j = 1, 2, \dots, m, \end{aligned}$$

then $v_1, v_2, \dots, v_m$ is an orthogonal system, and $w_1, w_2, \dots, w_m$ is an orthonormal system. Furthermore,

$$M = \text{span}(u_1, u_2, \dots, u_m) = \text{span}(v_1, v_2, \dots, v_m) = \text{span}(w_1, w_2, \dots, w_m).$$

PROOF (By induction) Suppose that

$$M_j = \text{span}(u_1, u_2, \dots, u_j) = \text{span}(v_1, v_2, \dots, v_j) = \text{span}(w_1, w_2, \dots, w_j),$$

where $v_1, \dots, v_j$ are orthogonal and $w_1, \dots, w_j$ are orthonormal. Now, for the induction step, assume that

$$v_{j+1} = u_{j+1} - \sum_{k=1}^j \frac{(u_{j+1}, v_k)v_k}{(v_k, v_k)}. \quad (4.2)$$

We have $v_{j+1} \notin M_j$ and $v_{j+1} \neq 0$ since u_{j+1} is independent of $u_1, u_2, \dots, u_j$. Also, $(v_{j+1}, v_\ell) = 0$ for $\ell = 1, 2, \dots, j$ (by simply plugging into (4.2)) and $(w_{j+1}, w_{j+1}) = 1$ since $w_{j+1} = v_{j+1}/\|v_{j+1}\|$. Thus, $v_1, \dots, v_{j+1}$ are orthogonal and $w_1, w_2, \dots, w_{j+1}$ are orthonormal. By construction,

$$\text{span}(w_1, w_2, \dots, w_{j+1}) = \text{span}(v_1, v_2, \dots, v_{j+1}) \subset \text{span}(u_1, u_2, \dots, u_{j+1}).$$

(Notice that by the induction hypothesis, each v_k is a linear combination of $u_1, u_2, \dots, u_j$ for $1 \leq k \leq j$.) Furthermore, we have

$$u_{j+1} = v_{j+1} + \sum_{k=1}^j \frac{(u_{j+1}, v_k)}{(v_k, v_k)} v_k.$$

Hence, $u_{j+1} \in \text{span}(v_1, v_2, \dots, v_{j+1})$. Therefore,

$$\begin{aligned} M_{j+1} &= \text{span}(u_1, u_2, \dots, u_{j+1}) = \text{span}(v_1, v_2, \dots, v_{j+1}) \\ &= \text{span}(w_1, w_2, \dots, w_{j+1}) \end{aligned}$$

for $j = 0, 1, \dots, m-1$. □

Example 4.6

(Two important cases)

(1) *Legendre Polynomials*

Let $V = C[-1, 1]$, $M = \text{span}(1, x, x^2)$,

$$(f, g) = \int_{-1}^1 f(x)g(x)dx \text{ for } f, g \in V, \text{ and } \|f\| = (f, f)^{\frac{1}{2}}.$$

Notice that $p \in M$ has the form $p(x) = a + bx + cx^2$. Using the Gram-Schmidt process,

$$\begin{aligned} v_1 &= 1, \\ w_1 &= \frac{1}{\left(\int_{-1}^1 dx\right)^{\frac{1}{2}}} = \frac{1}{\sqrt{2}}, \\ v_2 &= x - \frac{1}{2} \int_{-1}^1 x dx = x, \\ w_2 &= \frac{v_2}{\|v_2\|} = \frac{x}{\sqrt{2/3}}, \\ v_3 &= x^2 - \frac{1}{2} \int_{-1}^1 x^2 dx - \frac{x}{2/3} \int_{-1}^1 x^3 dx = x^2 - 1/3, \\ w_3 &= \frac{x^2 - 1/3}{\|x^2 - 1/3\|} = \frac{x^2 - 1/3}{\sqrt{8/45}}. \end{aligned}$$

Thus,

$$M = \text{span} \left(\frac{1}{\sqrt{2}}, \frac{x}{\sqrt{2/3}}, \frac{x^2 - 1/3}{\sqrt{8/45}} \right).$$

(Note: w_1, w_2 , and w_3 are the first three well-known *Legendre polynomials*. The entire sequence of Legendre polynomials is formed similarly.)

Now, we will find the best approximation to $f(x) = e^x \in V$ in M relative to the inner product

$$(u, v) = \int_{-1}^1 u(x)v(x)dx.$$

We need

$$Pf = \sum_{k=1}^3 (f, w_k) w_k.$$

Substituting the values of w_k we have just computed, we find that $(f, w_1) \approx 1.661985$, $(f, w_2) \approx 0.9011169$, and $(f, w_3) \approx 0.226302$. Thus,

$$\begin{aligned} e^x &\approx \frac{1.661985}{\sqrt{2}} + \frac{0.9011169x}{\sqrt{2/3}} + 0.226302 \left(\frac{x^2 - 1/3}{\sqrt{8/45}} \right) \\ &\approx .996293 + 1.103638x + 0.536722x^2. \end{aligned}$$

For comparison, the Taylor series about 0 gives $e^x \approx 1 + x + x^2/2$. See the following table.

x	Pf	Taylor series	e^x
-1	0.429377	0.500	0.367879
-1/2	0.578655	0.625	0.606531
0	0.996293	1.000	1.000000
1/2	1.682293	1.625	1.648721
1	2.636654	2.500	2.718282

(2) *Fourier Series*

Let $V = C[-\pi, \pi]$ with inner product

$$(f, g) = \int_{-\pi}^{\pi} f(x)g(x)dx \text{ and } \|f\| = (f, f)^{1/2}.$$

Using

$$\frac{1}{\pi} \int_{-\pi}^{\pi} \cos \ell x \cos m x dx = \delta_{\ell m}, \quad \frac{1}{\pi} \int_{-\pi}^{\pi} \sin \ell x \sin m x dx = \delta_{\ell m},$$

and

$$\frac{1}{\pi} \int_{-\pi}^{\pi} \cos \ell x \sin m x dx = 0,$$

it is readily seen that

$$\frac{1}{\sqrt{2\pi}}, \frac{\cos x}{\sqrt{\pi}}, \frac{\sin x}{\sqrt{\pi}}, \dots, \frac{\cos nx}{\sqrt{\pi}}, \frac{\sin nx}{\sqrt{\pi}}$$

are orthogonal. Let $M = \text{span} \left(\frac{1}{\sqrt{2\pi}}, \frac{\cos x}{\sqrt{\pi}}, \dots, \frac{\sin nx}{\sqrt{\pi}} \right)$ and let $f \in V$.

Then the best approximation in M to $f(x)$ is $w(x)$, where

$$w(x) = \frac{a_0}{\sqrt{2\pi}} + \sum_{\ell=1}^n \left(a_{\ell} \frac{\cos \ell x}{\sqrt{\pi}} + b_{\ell} \frac{\sin \ell x}{\sqrt{\pi}} \right),$$

and

$$a_0 = \left(f, \frac{1}{\sqrt{2\pi}} \right), \text{ and } a_{\ell} = \left(f, \frac{\cos \ell x}{\sqrt{\pi}} \right), \quad b_{\ell} = \left(f, \frac{\sin \ell x}{\sqrt{\pi}} \right),$$

for $\ell = 1, 2, \dots, n$.

□

Before continuing, it is worthwhile to summarize our study so far: If $W \subset V$ is a finite-dimensional subspace of a vector space V consisting of “nice” (easy to work with) elements, a fundamental problem in approximation theory is “given $v \in V$, find $w \in W$ close to v .”

We saw the following.

- (a) If V is a normed linear space, by Theorem 4.1, there exists a $w \in W$ such that $\|w - v\| \leq \|u - v\|$ for all $u \in W$. (However, as we will see, w may not be easy to find.)
- (b) If V is an inner product space, then the best approximation is easy to find, especially if an orthonormal basis is applied.

In the remainder of this chapter, we consider specific vector spaces V , such as $C[0, 1]$, along with various choices for spaces W of nice functions such as polynomials, trigonometric functions, and continuous piecewise polynomials. Furthermore, we consider approximations in different norms or normed linear spaces.

4.3 Polynomial Approximation

The simplest choice for W is a set of polynomials. Polynomials are easy to work with and understand. Furthermore, the following Weierstrass Approximation Theorem tells us that polynomials can provide good approximations.

4.3.1 The Weierstrass Approximation Theorem

The Weierstrass approximation theorem is

THEOREM 4.4

Given $f \in C[a, b]$ and $\epsilon > 0$ there exists a polynomial $p(x)$ such that

$$\|p - f\|_{\infty} = \max_{a \leq x \leq b} |f(x) - p(x)| < \epsilon.$$

REMARK 4.10 This result is somewhat surprising, because f is only required to be continuous, and polynomials are smooth. For example, the graph of $f(x)$ may be as in Figure 4.3, but a polynomial $p(x)$ can be found such that $\|f - p\|_{\infty} < \epsilon$, no matter how small we choose ϵ . Basically, we can make the change of direction of a polynomial's graph be arbitrarily abrupt by taking the degree of the polynomial to be sufficiently high. However, as we will see, the theorem is not practical for actually finding $p(x)$. $\square$

PROOF Let $x = (b - a)t + a$ for $t \in [0, 1]$. Then

$$h(t) = f((b - a)t + a)$$

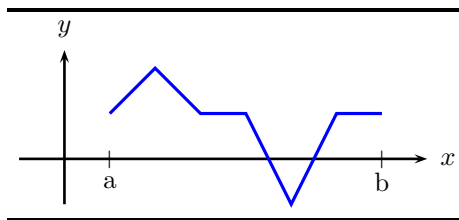


FIGURE 4.3: Graph of a nonsmooth but continuous $f(x)$; see Remark 4.10.

is continuous on $[0, 1]$. We can therefore restrict our attention to $h \in C[0, 1]$. To see this, observe that

$$|h(t) - p(t)| = \left| h\left(\frac{x-a}{b-a}\right) - p\left(\frac{x-a}{b-a}\right) \right| = \left| f(x) - p\left(\frac{x-a}{b-a}\right) \right|.$$

Define $B_m(h; t)$, $m = 1, 2, \dots$, by

$$B_m(h; t) = \sum_{k=0}^m h\left(\frac{k}{m}\right) \binom{m}{k} t^k (1-t)^{m-k}. \quad (4.3)$$

Clearly, $B_m(h; t) \in P_m$, where P_m is the set of polynomials of degree m or less. ($B_m(h; t)$ is called a *Bernstein polynomial* of degree m .) Furthermore, $B_m(h; t)$ can be regarded as a linear operator on $C[a, b]$, since

- (a) $B_m(\lambda h; t) = \lambda B_m(h; t)$
- (b) $B_m(h_1 + h_2; t) = B_m(h_1; t) + B_m(h_2; t)$.

In addition,

- (c) If $h_1(t) \leq h_2(t)$ for all $t \in [0, 1]$, then $B_m(h_1; t) \leq B_m(h_2; t)$ for all $t \in [0, 1]$.

We now show that, given $h \in C[0, 1]$ and $\epsilon > 0$, there exists an integer $m_0 \geq 0$ such that

$$\max_{0 \leq t \leq 1} |h(t) - B_{m_0}(h; t)| < \epsilon.$$

Hence, the theorem is proven by selecting

$$p(x) = B_{m_0}\left(h; \frac{x-a}{b-a}\right).$$

□

Before proving this, let's look at some special cases. (We also need these in the proof.) The special cases are

1. $h(t) = 1$,
2. $h(t) = t$, and
3. $h(t) = t^2$.

In detail,

$$1. B_m(1; t) = \sum_{k=0}^m \binom{m}{k} t^k (1-t)^{m-k} = (t + (1-t))^m = 1 \text{ for all } m \geq 1.$$

Thus, $B_m(1; t) = 1$ for $m \geq 1$.

$$\begin{aligned} 2. B_m(t; t) &= \sum_{k=0}^m \frac{k}{m} \binom{m}{k} t^k (1-t)^{m-k} \\ &= \sum_{k=0}^m \frac{k}{m} \frac{m!}{k!(m-k)!} t^k (1-t)^{m-k} \\ &= \sum_{k=0}^m \binom{m-1}{k-1} t^k (1-t)^{m-k} = \sum_{j=0}^{m-1} \binom{m-1}{j} t^{j+1} (1-t)^{m-j-1} \\ &= t \sum_{j=0}^{m-1} \binom{m-1}{j} t^j (1-t)^{m-1-j} = t(t + (1-t))^{m-1} = t \end{aligned}$$

for $m \geq 1$. Thus, $B_m(t; t) = t$ for $m \geq 1$.

$$\begin{aligned} 3. B_m(t^2; t) &= B_m(t^2 - t/m; t) + \frac{1}{m} B(t; t) \text{ (using linearity of } B_m(\cdot; t)) \\ &= B_m(t(t - 1/m); t) + t/m \\ &= \sum_{k=2}^m \frac{k}{m} \binom{k-1}{m} \binom{m}{k} t^k (1-t)^{m-k} + t/m \\ &= \sum_{k=2}^m \frac{k}{m} \frac{k-1}{m} \frac{m!}{k!(m-k)!} t^k (1-t)^{m-k} + t/m \\ &= \left(1 - \frac{1}{m}\right) \sum_{k=2}^m \binom{m-2}{k-2} t^k (1-t)^{m-k} + t/m. \end{aligned}$$

Hence,

$$\begin{aligned} B_m(t^2; t) &= t^2 \left(1 - \frac{1}{m}\right) \sum_{j=0}^{m-2} \binom{m-2}{j} t^j (1-t)^{m-2-j} + t/m \\ &= \left(1 - \frac{1}{m}\right) t^2 + t/m \\ &= t^2 + \frac{t(1-t)}{m}, \text{ for } m \geq 2. \end{aligned}$$

Thus,

$$B_m(t^2; t) = t^2 + \frac{t(1-t)}{m} \rightarrow t^2 \quad \text{as } m \rightarrow \infty$$

and

$$\|B_m(t^2; t) - t^2\|_\infty = \left\| \frac{t(1-t)}{m} \right\|_\infty = \frac{1/4}{m}.$$

Thus, for $\|B_m(t^2; t) - t^2\|_\infty \leq 10^{-7}$ requires $m \geq 2,500,000$.

Conclusion 4.1 *Bernstein polynomials are often not practical,³ but are useful theoretically.*

Now let us continue with the proof.

PROOF (continuation; we need to show that $\|B_m(h; t) - h\|_\infty < \epsilon$ for $h \in C[0, 1]$ for m sufficiently large.) Recall that $h \in C[0, 1]$. Suppose $\max_{0 \leq t \leq 1} |h(t)| = M$. Then

$$-2M \leq h(t) - h(s) \leq 2M \quad \text{for all } t, s \in [0, 1]. \quad (4.4)$$

Also, since $h(t)$ is continuous on $[0, 1]$, given $\epsilon_1 > 0$ there is a $\delta = \delta(\epsilon_1) > 0$ that does not depend on s or t (h is uniformly continuous) such that

$$|t - s| < \delta \quad \text{implies that} \quad -\epsilon_1 \leq h(t) - h(s) \leq \epsilon_1. \quad (4.5)$$

By (4.4) and (4.5), for any $s, t \in [0, 1]$,

$$-\epsilon_1 - \frac{2M}{\delta^2}(t-s)^2 \leq h(t) - h(s) \leq \epsilon_1 + \frac{2M}{\delta^2}(t-s)^2. \quad (4.6)$$

To see this, first suppose that $|t - s| < \delta$; then, (4.5) implies (4.6). Now suppose that $|t - s| \geq \delta$ so that $(t - s)^2/\delta^2 \geq 1$; then (4.4) implies (4.6). For the moment, fix $s \in [0, 1]$. Then (4.6) has the form $h_1(t) \leq h_2(t) \leq h_3(t)$, where

$$h_1(t) = -\epsilon_1 - \frac{2M}{\delta^2}(t-s)^2, \quad h_2(t) = h(t) - h(s), \quad \text{and} \quad h_3(t) = \epsilon_1 + \frac{2M}{\delta^2}(t-s)^2.$$

By linearity and monotonicity of $B_m(h; t)$ (see (a), (b), and (c) on page 206), we conclude that $B_m(h_1; t) \leq B_m(h_2; t) \leq B_m(h_3; t)$, so

$$-\epsilon_1 - \frac{2M}{\delta^2}B_m((t-s)^2; t) \leq B_m(h; t) - h(s) \leq \epsilon_1 + \frac{2M}{\delta^2}B_m((t-s)^2; t).$$

³with some exceptions. For example, in computer graphics rendering, high accuracy is not required to get a smooth picture, but the edges of the picture need to be smooth, and Bernstein approximations are extremely smooth.

But

$$\begin{aligned} B_m((t-s)^2; t) &= B_m(t^2; t) - 2sB_m(t; t) + s^2B_m(1; t) \\ &= t^2 - 2st + s^2 + \frac{t(1-t)}{m} = (t-s)^2 + \frac{t(1-t)}{m}. \end{aligned}$$

Thus,

$$|B_m(h; t) - h(s)| \leq \epsilon_1 + \left[(t-s)^2 + \frac{t(1-t)}{m} \right] \frac{2M}{\delta^2}.$$

Letting $t = s$, we have

$$|B_m(h; s) - h(s)| \leq \epsilon_1 + \frac{s(1-s)}{m} \frac{2M}{\delta^2} \leq \epsilon_1 + \frac{M}{2m\delta^2} \text{ for all } s \in [0, 1].$$

Choosing $\epsilon_1 = \epsilon/2$ and $m > M/(\delta^2\epsilon)$, we have $|B_m(h; s) - h(s)| < \epsilon$ for all $s \in [0, 1]$, where ϵ is arbitrarily small. $\square$

The Weierstrass Approximation Theorem tells us theoretically that polynomials can be good approximations to continuous functions. In the remainder of this section, we consider practical methods for obtaining highly accurate polynomial approximations.

4.3.2 Taylor Polynomial Approximations

Recall from Chapter 1 (Taylor's Theorem, on page 2) that if $f \in C^n[a, b]$ and $f^{(k+1)}(x)$ exists on $[a, b]$, then for $x_0 \in [a, b]$ there exists a $\xi(x)$ between x_0 and x such that $f(x) = P_n(x) + R_n(x)$, where

$$P_n(x) = \sum_{k=0}^n \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k,$$

and

$$R_n(x) = \int_{x_0}^x \frac{(x-t)^n}{n!} f^{(n+1)}(t) dt = \frac{f^{(n+1)}(\xi(x))(x - x_0)^{n+1}}{(n+1)!}.$$

$P_n(x)$ is the Taylor polynomial of $f(x)$ about $x = x_0$ and $R_n(x)$ is the remainder term. Taylor polynomials provide good approximations near $x = x_0$. However, away from $x = x_0$, Taylor polynomials can be poor approximations. In addition, Taylor series require smooth functions. Nonetheless, automatic differentiation techniques, as explained in Section 6.2 on page 327, can be used to obtain high-order derivatives for complicated but smooth functions.

4.3.3 Lagrange Interpolation

Problem: Given $n + 1$ distinct real numbers $x_0, x_1, x_2, \dots, x_n$ and $n + 1$ arbitrary numbers $y_0, y_1, \dots, y_n$, find the polynomial of degree at most n such that $y_j = p(x_j)$ for $j = 0, 1, 2, \dots, n$.

We will address this problem in this section.

THEOREM 4.5

For any $n + 1$ distinct real numbers $x_0, x_1, \dots, x_n$ and for arbitrary real numbers $y_0, y_1, \dots, y_n$, there exists a unique interpolating polynomial of degree at most n such that $p(x_j) = y_j$, $j = 0, 1, \dots, n$.

PROOF We first define a useful set of polynomials of degree n denoted by $\ell_0, \ell_1, \dots, \ell_n$ for points $x_0, x_1, \dots, x_n \in \mathbb{R}$ as

$$\ell_k(x) = \prod_{\substack{i=0 \\ i \neq k}}^n \frac{x - x_i}{x_k - x_i}, \quad k = 0, 1, \dots, n. \quad (4.7)$$

Notice that

(i) $\ell_k(x)$ is of degree n for each $k = 0, 1, \dots, n$.

(ii) $\ell_k(x_j) = \begin{cases} 0 & \text{if } j \neq k, \\ 1 & \text{if } j = k, \end{cases}$
that is, $\ell_k(x_j) = \delta_{kj}$.

Now let

$$p(x) = \sum_{k=0}^n y_k \ell_k(x).$$

Then

$$p(x_j) = \sum_{k=0}^n y_k \ell_k(x_j) = \sum_{k=0}^n y_k \delta_{jk} = y_j \quad \text{for } j = 0, 1, \dots, n.$$

Thus,

$$p(x) = \sum_{k=0}^n y_k \ell_k(x)$$

is a polynomial of degree at most n that passes through points (x_j, y_j) , $j = 0, 1, 2, \dots, n$. This is called the *Lagrange form* of the interpolating polynomial, and the set of functions $\{\ell_k\}_{k=0}^n$ is called the *Lagrange basis* for the space of polynomials of degree n associated with the set of points $\{x_i\}_{i=0}^n$.

We now show that the polynomial is unique. Suppose that $q(x) \neq p(x)$, $q(x_j) = p(x_j) = y_j$, $j = 0, 1, 2, \dots, n$, and q and p are polynomials of degree at most n . Let $p(x) = \sum_{k=0}^n \alpha_k x^k$ and $q(x) = \sum_{k=0}^n \beta_k x^k$. Then

$$p(x) - q(x) = \sum_{k=0}^n (\alpha_k - \beta_k) x^k = \sum_{k=0}^n \gamma_k x^k.$$

(We will show that $\gamma_k = 0$ for $k = 0, 1, 2, \dots, n$ and thus, $p(x) = q(x)$.) Since $p(x_j) = q(x_j)$,

$$\sum_{k=0}^n \gamma_k x_j^k = 0$$

for $j = 0, 1, 2, \dots, n$. This can be expressed as the linear system

$$A\gamma = \begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & & & x_1^n \\ & & \vdots & & \ddots & \\ 1 & x_n & x_n^2 & & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

The above matrix A is called the *Vandermonde matrix*. As shown in Remark 4.11 below,

$$\det(A) = \prod_{k=1}^n \prod_{j=0}^{k-1} (x_k - x_j) \neq 0,$$

since $x_k \neq x_j$ for $j \neq k$. Hence, A is nonsingular, so $\gamma_0 = \gamma_1 = \cdots = \gamma_n = 0$. $\square$

REMARK 4.11 Consider

$$V_n(x) = \det \begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & & & x_1^n \\ & & \vdots & & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & & x_{n-1}^n \\ 1 & x & x^2 & \cdots & & x^n \end{pmatrix} = a_0 + a_1x + \cdots + a_nx^n$$

(a polynomial of degree n). But

$$V_n(x) = a_n(x - x_0)(x - x_1) \cdots (x - x_{n-1}),$$

since the zeros of $V_n(x)$ are $x_0, \dots, x_{n-1}$. (To see this, note that two rows of the determinant are identical when x is replaced by x_i .) Furthermore, observing that $a_n = V_{n-1}(x_{n-1})$ shows us that

$$V_n(x) = V_{n-1}(x_{n-1})(x - x_0) \cdots (x - x_{n-1}) = V_{n-1}(x_{n-1}) \prod_{j=0}^{n-1} (x - x_j).$$

(There are various ways of seeing this, involving subtracting multiples of one row from another, factoring out common factors from rows or columns, and expansion by minors.) Therefore, by observing $V_1(x_1) = (x_1 - x_0)$ and using

an induction argument (or else repeating the above process with $V_{n-1}(x)$ in place of $V_n(x)$), we obtain

$$\det(A) = V_n(x_n) = \prod_{k=1}^n \prod_{j=0}^{k-1} (x_k - x_j),$$

where A is the above Vandermonde matrix. $\square$

Summarizing, we obtain the *Lagrange form* of the (unique) interpolating polynomial:

$$p(x) = \sum_{k=0}^n y_k \ell_k(x) \quad \text{with} \quad \ell_k(x) = \prod_{\substack{j=0 \\ j \neq k}}^n \frac{x - x_j}{x_k - x_j}. \quad (4.8)$$

where $\{\ell_k\}_{k=0}^n$ is the *Lagrange basis* for the space of polynomials of degree n associated with the set of points $\{x_i\}_{i=0}^n$, and we will call the ℓ_k the *Lagrange basis functions*.

REMARK 4.12 An important feature of the Lagrange basis is that it is *collocating*. The salient property of a collocating basis is that the matrix of the system of equations to be solved for the coefficients is the identity matrix. That is, the matrix in the system of equations

$$\{p(x_i) = y_i\}_{i=0}^n$$

to solve for the c_i in the representation

$$p(x) = \sum_{k=0}^n c_k \ell_k(x)$$

is

$$\begin{pmatrix} \ell_0(x_0) & \ell_1(x_0) & \cdots & \ell_n(x_0) \\ \ell_0(x_1) & \ell_1(x_1) & \cdots & \ell_n(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \ell_0(x_n) & \ell_1(x_n) & \cdots & \ell_n(x_n) \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix} \quad (4.9)$$

is the identity matrix. (Contrast this to the Vandermonde matrix, where we use x^k instead of $\ell_k(x)$. The Vandermonde matrix becomes ill-conditioned for n moderately sized, while the identity matrix is perfectly conditioned; indeed, we need do no work to solve (4.9).) $\square$

Although the interpolating polynomial is unique, there are various ways of representing it, just as there are various bases that can be used to represent a vector in $(n+1)$ -dimensional space. In the next section, we consider representing the interpolating polynomial in the so-called Newton form.

4.3.4 The Newton Form for the Interpolating Polynomial

Although the Lagrange polynomials form a collocating basis, useful in theory for symbolically deriving formulas such as for numerical integration, the Lagrange representation (4.8) is generally not used to numerically evaluate interpolating polynomials. This is because

- (1) it requires many operations to evaluate $p(x)$ for many different values of x , and
- (2) all ℓ_k 's change if another point (x_{n+1}, y_{n+1}) is added.

These problems are alleviated in the Newton form of the interpolating polynomial. To describe this form we use the following.

DEFINITION 4.6 $y[x_j, x_{j+1}] = \frac{y_{j+1} - y_j}{x_{j+1} - x_j}$ is the first divided difference.

DEFINITION 4.7

$$y[x_j, x_{j+1}, \dots, x_{j+k}] = \frac{y[x_{j+1}, \dots, x_{j+k}] - y[x_j, \dots, x_{j+k-1}]}{x_{j+k} - x_j}$$

is the k -th order divided difference (and is defined iteratively).

Consider first the linear interpolant through (x_0, y_0) , and (x_1, y_1) :

$$p_1(x) = y_0 + (x - x_0)y[x_0, x_1],$$

since $p_1(x)$ is of degree 1, $p_1(x_0) = y_0$, and

$$p_1(x_1) = y_0 + (x_1 - x_0)\frac{y_1 - y_0}{x_1 - x_0} = y_1.$$

Consider now the quadratic interpolant through (x_0, y_0) , (x_1, y_1) , and (x_2, y_2) . We have

$$p_2(x) = p_1(x) + (x - x_0)(x - x_1)y[x_0, x_1, x_2],$$

since $p_2(x)$ is of degree 2, $p_2(x_0) = y_0$, $p_2(x_1) = y_1$, and

$$\begin{aligned} p_2(x_2) &= p_1(x_2) + (x_2 - x_0)(x_2 - x_1)\frac{y[x_1, x_2] - y[x_0, x_1]}{x_2 - x_0} \\ &= y_0 + (x_2 - x_0)\frac{y_1 - y_0}{x_1 - x_0} + (x_2 - x_1)\frac{y_2 - y_1}{x_2 - x_1} - (x_2 - x_1)\frac{y_1 - y_0}{x_1 - x_0} \\ &= y_0 + y_2 - y_1 \\ &\quad + \frac{1}{x_1 - x_0}[(x_2 - x_0 - x_2 + x_1)y_1 + (-x_2 + x_0 - x_1 + x_2)y_0] \\ &= y_2. \end{aligned}$$

Continuing this process, one obtains:

$$\begin{aligned} p_n(x) &= p_{n-1}(x) + (x - x_0)(x - x_1) \dots (x - x_{n-1})y[x_0, x_1, \dots, x_n] \\ &= y_0 + (x - x_0)y[x_0, x_1] + (x - x_0)(x - x_1)y[x_0, x_1, x_2] + \dots \\ &\quad + \left\{ \prod_{i=0}^{n-1} (x - x_i) \right\} y[x_0, \dots, x_n]. \end{aligned}$$

This is called *Newton's divided-difference formula* for the interpolating polynomial through the points $\{(x_j, y_j)\}_{j=0}^n$. This is a computationally efficient form, because the divided differences can be rapidly calculated using the following tabular arrangement, which is easily implemented on a computer.

j	x_j	y_j	$y[x_j, x_{j+1}]$	$y[x_j, x_{j+1}, x_{j+2}]$	$y[x_j, x_{j+1}, x_{j+2}, x_{j+3}]$
0	x_0	y_0			
1	x_1	y_1	$\frac{y_1 - y_0}{x_1 - x_0} = y[x_0, x_1]$		
2	x_2	y_2	$\frac{y_2 - y_1}{x_2 - x_1} = y[x_1, x_2]$	$\frac{y[x_1, x_2] - y[x_0, x_1]}{x_2 - x_0} = y[x_0, x_1, x_2]$	
3	x_3	y_3	$\frac{y_3 - y_2}{x_3 - x_2} = y[x_2, x_3]$	$\frac{y[x_2, x_3] - y[x_1, x_2]}{x_3 - x_1} = y[x_1, x_2, x_3]$	$\frac{y[x_1, x_2, x_3] - y[x_0, x_1, x_2]}{x_3 - x_0} = y[x_0, x_1, x_2, x_3]$
4	x_4	y_4	$\frac{y_4 - y_3}{x_4 - x_3} = y[x_3, x_4]$	$\frac{y[x_3, x_4] - y[x_2, x_3]}{x_4 - x_2} = y[x_2, x_3, x_4]$	$\frac{y[x_2, x_3, x_4] - y[x_1, x_2, x_3]}{x_4 - x_1} = y[x_1, x_2, x_3, x_4]$

Example 4.7

Consider $y(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} dx$ (standard normal distribution)

j	x_j	y_j	$y[x_j, x_{j+1}]$	$y[x_j, x_{j+1}, x_{j+2}]$	$y[x_j, x_{j+1}, x_{j+2}, x_{j+3}]$
0	1.4	0.9192			
1	1.6	0.9452	$\frac{0.9452 - 0.9192}{0.2} = 0.130$		
2	1.8	0.9641	0.0945	$\frac{0.0945 - 0.130}{0.4} = -0.08875$	
3	2.0	0.9772	0.0655	-0.0725	$\frac{-0.0725 + 0.08875}{0.6} = 0.02708$

Thus,

$$p_1(x) = 0.9192 + (x - 1.4)0.130$$

is the line through $(1.4, 0.9192)$ and $(1.6, 0.9452)$. Hence, $y(1.65) \approx p_1(1.65) \approx 0.9517$. Also,

$$p_2(x) = 0.9192 + (x - 1.4)0.130 + (x - 1.4)(x - 1.6)(-0.08875)$$

is a quadratic polynomial through (x_0, y_0) , (x_1, y_1) , and (x_2, y_2) . Hence, $y(1.65) \approx p_2(1.65) \approx 0.9506$. Finally,

$$p_3(x) = p_2(x) + (x - 1.4)(x - 1.6)(x - 1.8)(0.027083)$$

is the cubic polynomial through all four points and $y(1.65) \approx p_3(1.65) \approx 0.9505$, which is accurate to four digits. $\square$

REMARK 4.13 If the points are equally spaced, i.e., $x_{j+1} - x_j = \Delta x$ for all j , Newton's divided difference formula can be simplified. (See, e.g., [6].) The resulting formula is called *Newton's forward difference formula*. If the points $x_0, \dots, x_n$ are reordered to $x_n, x_{n-1}, \dots, x_0$, the resulting formula is called *Newton's backward difference formula*. $\square$

REMARK 4.14 The functions

$$N_k(x) = \prod_{i=0}^k (x - x_i), \quad k = 1, \dots, n \quad (4.10)$$

form a basis for the space of polynomials of degree n . In contrast to the Lagrange basis, this basis and the conditions $\{p(x_i) = y_i\}_{i=0}^n$ do not lead to a linear system of equations in which the matrix is the identity matrix, but do lead to a lower triangular system of equations. You will show this in Exercise 16 in this chapter. $\square$

4.3.4.1 An Error Formula for the Interpolating Polynomial

We now consider the error in approximating a given function $f(x)$ by an interpolating polynomial $p(x)$ that passes through the $n+1$ points $(x_j, f(x_j))$, $j = 0, 1, 2, \dots, n$.

THEOREM 4.6

If $x_0, x_1, \dots, x_n$ are $n+1$ distinct points in $[a, b]$ and $f \in C^{n+1}[a, b]$, then for each $x \in [a, b]$, there exists a number $\xi = \xi(x) \in (a, b)$ such that

$$f(x) = p(x) + \frac{f^{(n+1)}(\xi(x)) \prod_{j=0}^n (x - x_j)}{(n+1)!} \quad (4.11)$$

PROOF

Case 1 Suppose that $x = x_k$ for some k , $0 \leq k \leq n$. Then, $f(x_k) = p(x_k)$ and (4.11) is satisfied for $\xi(x_k)$ arbitrary on (a, b) .

Case 2 Suppose that $x \neq x_k$ for $k = 0, 1, 2, \dots, n$. Define as a function of t ,

$$g(t) = f(t) - p(t) - (f(x) - p(x)) \prod_{i=0}^n \frac{t - x_i}{x - x_i}.$$

Since $f \in C^{n+1}[a, b]$, $p \in C^\infty[a, b]$, and $x \neq x_i$ for any i , it follows that $g \in C^{n+1}[a, b]$. For $t = x_k$,

$$g(x_k) = f(x_k) - p(x_k) - (f(x) - p(x)) \prod_{i=0}^n \frac{x_k - x_i}{x - x_i} = 0 - 0 = 0.$$

Moreover,

$$g(x) = f(x) - p(x) - (f(x) - p(x)) \prod_{i=0}^n \frac{x - x_i}{x - x_i} = 0.$$

Thus, $g \in C^{n+1}[a, b]$ vanishes at the $(n+2)$ points $x_0, x_1, \dots, x_n, x$. If $g(x_0) = 0$ and $g(x_1) = 0$, there is an x_0^* , $x_0 < x_0^* < x_1$, such that $g'(x_0^*) = 0$. (This is called *Rolle's Theorem*, a special case of the Intermediate Value Theorem as stated on page 1.) Thus, $g'(t)$ vanishes $n+1$ times. Continuing this argument, $g^{(n+1)}(t)$ vanishes at least once on (a, b) . (This is called the *Generalized Rolle's theorem*.) Thus, there exists a $\xi = \xi(x)$ in (a, b) for which $g^{(n+1)}(\xi) = 0$. Evaluating $g^{(n+1)}(t)$ at ξ gives

$$0 = g^{(n+1)}(\xi) = f^{(n+1)}(\xi) - p^{(n+1)}(\xi) - [f(x) - p(x)] \left(\frac{d^{n+1}}{dt^{n+1}} \prod_{i=0}^n \frac{t - x_i}{x - x_i} \right)_{t=\xi},$$

so

$$0 = f^{(n+1)}(\xi) - (f(x) - p(x)) \frac{(n+1)!}{\prod_{i=0}^n (x - x_i)}.$$

Therefore,

$$f(x) = p(x) + \frac{f^{(n+1)}(\xi(x)) \prod_{i=0}^n (x - x_i)}{(n+1)!}.$$

□

In the next section, we will see yet another basis for the set of polynomials of degree n that is useful in analyzing the error (4.11) and reducing it.

4.3.4.2 Optimal Points of Interpolation: Chebyshev Polynomials

Now consider the error estimate (4.11). We have

$$\|f - p\|_{\infty} = \max_{a \leq x \leq b} |f(x) - p(x)| \leq \frac{1}{(n+1)!} \|f^{(n+1)}\|_{\infty} \|W\|_{\infty}, \quad (4.12)$$

where

$$W(x) = \prod_{j=0}^n (x - x_j).$$

We see that the error bound depends on the nodes $\{x_j\}_{j=0}^n$ through $\|W\|_{\infty}$. We thus pose the following question: Can we choose $\{x_i\}_{i=0}^n$ suitably so that $\|W\|_{\infty}$ is minimized? We first will consider the interval $[a, b] = [-1, 1]$.

THEOREM 4.7

The uniform norm of $W(x) = \prod_{i=0}^n (x - x_i)$ is minimized on $[-1, 1]$ when

$$x_i = \cos \left(\frac{2i+1}{n+1} \cdot \frac{\pi}{2} \right) \quad 0 \leq i \leq n,$$

and the minimum value of the norm is $\|W\|_\infty = 2^{-n}$.

PROOF We make a change of variables on $[-1, 1]$, letting $x = \cos \theta$ and restricting θ to $[0, \pi]$. Consider the trigonometric identity

$$\cos((k+1)\theta) = 2 \cos(\theta) \cos(k\theta) - \cos((k-1)\theta), \quad k = 0, 1, 2, \dots, \quad (4.13)$$

and define for $\theta \in [0, \pi]$,

$$T_k(x) = \cos(k\theta), \quad \theta = \arccos x, \quad k = 0, 1, 2, \dots \quad (4.14)$$

In terms of the $T_k(x)$'s, (4.13) gives the recursion relation

$$T_{k+1}(x) = 2T_1(x)T_k(x) - T_{k-1}(x), \quad k = 1, 2, 3, \dots \quad (4.15)$$

Also, by (4.14), $T_0(x) = 1$ and $T_1(x) = x$, so (4.15) gives $T_2(x) = 2x^2 - 1$, $T_3(x) = 4x^3 - 3x$, $T_4(x) = 8x^4 - 8x^2 + 1$, ... By induction, we can conclude

(a) $T_k(x)$ is a polynomial in x of degree k ;

(b) the leading term of $T_k(x)$ is $2^{k-1}x^k$.

The polynomials $T_k(x)$ are called *Chebyshev* (or *Tschebycheff*).⁴ By (4.14), we see for $k \geq 1$, $T_k(x) = 0$ implies that $\cos(k\theta) = 0$ with $0 \leq k\theta \leq k\pi$. Hence, $k\theta = (2i-1)\pi/2$, $1 \leq i \leq k$, i.e.,

$$\theta = \frac{2i-1}{k} \cdot \frac{\pi}{2}, \quad 1 \leq i \leq k.$$

Thus, the zeros of the k -th Chebyshev polynomial $T_k(x)$ are

$$x_i^{(k)} = \cos \left(\frac{2i+1}{k} \cdot \frac{\pi}{2} \right), \quad 0 \leq i \leq k-1. \quad (4.16)$$

We now consider the polynomial $\tilde{W}(x) = 2^{-n}T_{n+1}(x)$ on $[-1, 1]$. By (4.16), the zeros of $\tilde{W}$ are

$$x_i = \cos \left(\frac{2i+1}{n+1} \cdot \frac{\pi}{2} \right), \quad 0 \leq i \leq n.$$

⁴The original spelling uses the Cyrillic alphabet.

Also, since $\tilde{W}(x)$ is a polynomial of degree $n + 1$ with roots x_i , $0 \leq i \leq n$, it must be equal to $W(x) = \prod_{i=0}^n (x - x_i)$, where W is as in the statement of the theorem. Hence,

$$W(x) = 2^{-n} T_{n+1}(x). \quad (4.17)$$

Now, the maximum value of $|W(x)|$ on $[-1, 1]$ occurs at the $n + 2$ points z_i where $T_{n+1}(z_i) = \pm 1$. By (4.14), these points are

$$z_i = \cos\left(\frac{i\pi}{n+1}\right), \quad i = 0, 1, 2, \dots, n+1.$$

Therefore, $\|W\|_\infty = 2^{-n}$. Now consider $V(x)$ of the same form as $W(x)$ but $\|V\|_\infty < \|W\|_\infty$, i.e.,

$$V(x) = \prod_{i=0}^n (x - \hat{x}_i),$$

$\hat{x}_i \neq x_i$ for at least one i . Notice that $W(z_i) = (-1)^i 2^{-n}$. Thus, as is depicted graphically in Figure 4.4, $W - V$ must alternate in sign at each of the points

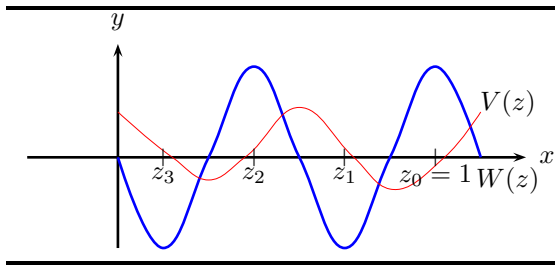


FIGURE 4.4: The Chebyshev equi-oscillation property (Theorem 4.7).

$z_0, z_1, z_2, \dots, z_{n+1}$, so, by Rolle's theorem, $W - V$ must have a root in each of these $n + 1$ intervals. However, since both V and W are *monic*,⁵ $W - V$ is a polynomial of degree n or less. Therefore, $W - V = 0$, that is, $W(x) = V(x)$, thus contradicting $\|V\|_\infty < \|W\|_\infty$. We conclude that W minimizes the maximum deviation from 0 among all polynomials of degree $n + 1$. $\square$

REMARK 4.15 If the interval of approximation is $[a, b]$ rather than $[-1, 1]$, we use the transformation

$$x = \frac{a - 2y + b}{a - b}$$

⁵that is, since the leading coefficient of each is 1

to map $[a, b]$ to $[-1, 1]$. We now approximate

$$g(x) = f\left(\frac{1}{2}(b-a)x + \frac{1}{2}(a+b)\right)$$

for $-1 \leq x \leq 1$. Thus, the corresponding “optimal” interpolation points in $[a, b]$ are

$$y_i = \frac{1}{2}[(b-a)x_i + (a+b)], \quad 0 \leq i \leq n,$$

where the x_i are given in Theorem 4.7. In this case, one can show that

$$\|W\|_\infty = 2^{-2n-1}(b-a)^{n+1}$$

(Exercise 7 on page 285). □

REMARK 4.16 By Theorems 4.6 and 4.7, using the zeros of the Chebyshev polynomial as the interpolating nodes, one obtains

$$\|f - p\|_\infty \leq \frac{1}{(n+1)!} \|f^{(n+1)}\|_\infty (b-a)^{n+1} 2^{-2n-1}. \quad (4.18)$$

Compare this to the general error bound: for any set of distinct nodes, $\|W\|_\infty \leq (b-a)^{n+1}$, so one always has

$$\|f - p\|_\infty \leq \frac{1}{(n+1)!} \|f^{(n+1)}\|_\infty (b-a)^{n+1}.$$

□

REMARK 4.17 Consider now fixing $[a, b]$ and letting $n \rightarrow \infty$. One would expect $\|f - p_n\|_\infty \rightarrow 0$ as $n \rightarrow \infty$ where p_n is the interpolating polynomial of degree n . However, this is not true in general. For example, consider *Runge’s function*:

$$f(x) = \frac{1}{1+x^2} \in C^\infty[-5, 5].$$

If we use equally spaced nodes on $[-5, 5]$, it can be shown that $\|p_n - f\|_\infty \rightarrow \infty$ as $n \rightarrow \infty$. However, if the nodes are the Chebyshev points and $f \in C^2[-1, 1]$, then $\|f - p_n\| \rightarrow 0$ as $n \rightarrow \infty$. (In fact, $\|f - p_n\|_\infty = \mathcal{O}(1/\sqrt{n})$ as $n \rightarrow \infty$.) Nevertheless, in the general case of $f \in C[-1, 1]$, it can be shown no matter how the interpolating points are distributed, there exists a continuous function f for which $\|p_n - f\|_\infty \rightarrow \infty$ as $n \rightarrow \infty$. □

This observation motivates us to consider other methods for approximating functions using polynomials, in particular piecewise polynomial interpolation. We will do that later. First, we briefly consider interpolation in which we

specify not only function values, but both function and derivative values at points.⁶ A simple scheme for doing so is Hermite interpolation.

4.3.5 Hermite Interpolation

In Hermite interpolation, we find the polynomial $p(x)$ of degree at most $2n - 1$ that interpolates the data⁷ $(x_i, f(x_i))$, $(x_i, f'(x_i))$ for $i = 1, 2, \dots, n$, where $p(x_i) = f(x_i)$ and $p'(x_i) = f'(x_i)$ for $i = 1, 2, \dots, n$, and the x_i are distinct points. We have the following.

THEOREM 4.8

If $x_1, x_2, \dots, x_n \in [a, b]$ are distinct points and $f \in C^1[a, b]$, the unique polynomial of degree at most $2n - 1$ that agrees with $f(x)$ and $f'(x)$ at $x_1, x_2, \dots, x_n$ is given by

$$H_n(x) = \sum_{k=1}^n f(x_k) h_k(x) + \sum_{k=1}^n f'(x_k) \tilde{h}_k(x),$$

where

$$h_k(x) = [1 - 2\ell'_k(x_k)(x - x_k)](\ell_k(x))^2, \quad \text{and} \quad \tilde{h}_k(x) = (x - x_k)(\ell_k(x))^2,$$

where

$$\ell_k(x) = \frac{\Psi_n(x)}{(x - x_k)\Psi'_n(x_k)} = \prod_{\substack{j=1 \\ j \neq k}}^n \frac{x - x_j}{x_k - x_j}, \quad \text{where} \quad \Psi_n(x) = \prod_{k=1}^n (x - x_k).$$

Moreover, if $f \in C^{2n}[a, b]$, then

$$f(x) - H_n(x) = \frac{(\Psi_n(x))^2}{(2n)!} f^{2n}(\xi) \quad \text{for some} \quad \xi = \xi(x) \in [a, b]. \quad (4.19)$$

$H_n(x)$ is called the Hermite interpolating polynomial.

REMARK 4.18 The ℓ_k in Theorem 4.8 are the same as the Lagrange basis functions we saw on page 212. $\square$

⁶One observes that the interpolating polynomial to Runge's function oscillates wildly as the degree increases. Specifying the derivative at points conceivably will limit rapid increases and decreases.

⁷Here, it is more convenient to start the indexing with 1, rather than 0.

PROOF Note that

$$\begin{aligned} h'_k(x_j) &= \tilde{h}_k(x_j) = 0 \quad \text{for } 1 \leq k, j \leq n, \quad \text{and} \\ h_k(x_j) &= \tilde{h}'_k(x_j) = \begin{cases} 1 & k = j, \\ 0 & k \neq j. \end{cases} \end{aligned}$$

Thus, $H_n(x_j) = f(x_j)$ and $H'_n(x_j) = f'(x_j)$ for $j = 1, 2, \dots, n$, so $H_n(x)$ satisfies the interpolation requirements. To show uniqueness, suppose $G(x)$ of degree less than or equal to $2n - 1$ also satisfies the interpolation conditions. Let

$$R(x) = H_n(x) - G(x).$$

Then $R(x_i) = R'(x_i) = 0$, so R has double roots at $x_1, \dots, x_n$. Hence,

$$R(x) = q(x)(x - x_1)^2(x - x_2)^2 \dots (x - x_n)^2$$

for some polynomial $q(x)$. If $q(x) \neq 0$, then $R(x)$ has degree greater than or equal to $2n$, which is a contradiction. Thus, $q(x) = 0$, so $R(x) = 0$, so the Hermite interpolating polynomial must be unique.

Now consider (4.19). The formula is trivially satisfied for $x = x_i$ for any i . Suppose that $x \neq x_i$ for any i . Let

$$g(t) = f(t) - H_n(t) - \frac{(t - x_1)^2 \dots (t - x_n)^2}{(x - x_1)^2 \dots (x - x_n)^2} (f(x) - H_n(x)).$$

Then $g(x_i) = 0$ for each i and $g(x) = 0$, so Rolle's Theorem implies that $g'(t)$ has n distinct zeros ξ_j , $1 \leq j \leq n$ on $[a, b]$, with $\xi_j \neq x_j$ for any j . Examining

$$g'(t) = f'(t) - H'_n(t) - \frac{d}{dt} \left\{ \frac{(t - x_1)^2 \dots (t - x_n)^2}{(x - x_1)^2 \dots (x - x_n)^2} (f(x) - H_n(x)) \right\},$$

we see that $g'(x_i) = 0$ for $i = 1, 2, \dots, n$. Therefore, $g'(t)$ has $2n$ distinct zeros on $[a, b]$. By the Generalized Rolle's Theorem, $g^{(2n)}(t)$ has at least one zero $\xi(x) \in [a, b]$. Thus,

$$g^{(2n)}(\xi(x)) = 0 = f^{(2n)}(\xi) - 0 - \frac{2n!}{\prod_{j=0}^n (x - x_j)^2} (f(x) - H_n(x)),$$

so

$$f(x) - H_n(x) = \frac{(\Psi_n(x))^2}{(2n)!} f^{(2n)}(\xi(x)).$$

□

In the next section, we consider approximation by polynomials in such a way that the graph does not necessarily go through the data exactly. This type of approximation is appropriate, for example, when there is much data, and the data contains small errors, or when we need to approximate an underlying function with a low-degree polynomial.

4.3.6 Least Squares Polynomial Approximation

We introduced the least squares problem in Section 3.3.8 on page 139, where we saw that a QR decomposition was a stable way of computing solutions. We also saw, in Section 3.5 (page 174), that least squares solutions to discrete problems could be analyzed with the singular value decomposition. Here, we will examine the least squares problem in the context of spaces of functions and orthogonal polynomials.

Recall the general least squares problem:

DEFINITION 4.8 *Let V be an inner product space and let $W \subset V$ be a finite-dimensional subspace of V . Suppose*

$$W = \text{span}(w_1, w_2, \dots, w_n),$$

and suppose that $v \in V$. The general least squares problem is to find $w \in W$ such that

$$\|w - v\|^2 \leq \|u - v\|^2 \quad \text{for all } u \in W,$$

where $\|f\| = (f, f)^{\frac{1}{2}}$. That is, we minimize

$$F(\alpha_1, \alpha_2, \dots, \alpha_n) = \left(\sum_{j=1}^n \alpha_j w_j - v, \sum_{k=1}^n \alpha_k w_k - v \right),$$

where

$$w = \sum_{j=1}^n \alpha_j w_j.$$

We saw that we could find $\alpha_1, \alpha_2, \dots, \alpha_n$ by solving the linear system

$$\sum_{j=1}^n \alpha_j (w_j, w_\ell) = (v, w_\ell), \quad \ell = 1, 2, \dots, n. \quad (4.20)$$

That is, $A\alpha = b$ where

$$A\alpha = \begin{pmatrix} (w_1, w_1) & (w_1, w_2) & \cdots & (w_1, w_n) \\ (w_2, w_1) & & & \vdots \\ \vdots & & \ddots & \\ (w_n, w_1) & \cdots & & (w_n, w_n) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} (v, w_1) \\ (v, w_2) \\ \vdots \\ (v, w_n) \end{pmatrix} = b. \quad (4.21)$$

We have previously seen (4.21) both in Section 3.3.8, where we saw how to solve (4.21) with a QR factorization in the case of finite-dimensional Hilbert

spaces, and in Section 4.2.2 (on page 198). There, we mentioned that A is positive definite and thus nonsingular. (You will show this in Problem 8 on page 285 below.) Thus, α is uniquely determined. We also saw that $\{w_1, \dots, w_n\}$ could be made orthogonal by the Gram–Schmidt procedure. Suppose that $\{w_1^*, \dots, w_n^*\}$ are orthogonal. Then (4.20) becomes

$$\alpha_j(w_j^*, w_j^*) = (v, w_j^*), \text{ or } \alpha_j = \frac{(v, w_j^*)}{(w_j^*, w_j^*)}, \quad (4.22)$$

and

$$w = \sum_{j=1}^n \frac{(v, w_j^*) w_j^*}{(w_j^*, w_j^*)}$$

is the solution to the least squares problem.

In this section, we consider $V = C[a, b]$ and $W = P^n$, the space of polynomials of degree less than or equal to n with the inner product

$$(f, g) = \int_a^b f(x)g(x)\rho(x)dx, \text{ with } \rho(x) > 0 \text{ for } x \in (a, b).$$

Thus, $W = \text{span}(1, x, x^2, \dots, x^n)$.

REMARK 4.19 Generally, to find the least squares approximation, an orthogonal set of polynomials is first found with respect to weight function $\rho(x)$. The reason is that A in (4.20) can be very ill-conditioned. Consider, for example, $a = 0$, $b = 1$, and $\rho(x) = 1$. Then the matrix A has the form

$$A = \begin{pmatrix} (1, 1) & (1, x) & \cdots & (1, x^n) \\ (x, 1) & (x, x) & \cdots & (x, x^n) \\ & & \ddots & \\ (x^n, 1) & \cdots & & (x^n, x^n) \end{pmatrix} = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n+1} \\ \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n+2} \\ & & \ddots & \\ \frac{1}{n+1} & \cdots & & \frac{1}{2n+1} \end{pmatrix},$$

which is a Hilbert matrix. (We saw that Hilbert matrices are very ill-conditioned; see Table 3.1 on page 125.) $\square$

REMARK 4.20 We give intervals, weight functions, and recurrence relations for several classical sets of orthogonal polynomials. Given a , b , and $\rho(x)$, the set of orthogonal polynomials can be obtained using the Gram–Schmidt process. However, we will see later that any set of orthogonal polynomials also obeys a three-term recurrence relation.

(a) *Legendre polynomials* $P_n(x)$

$a = -1$, $b = 1$, $\rho(x) = 1$, $P_0(x) = 1$, $P_1(x) = x$

Recurrence relation:

$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x), \quad n \geq 1.$$

(b) *Chebyshev* (Tshebycheff) Polynomials $T_n(x)$

$$a = -1, b = 1, \rho(x) = 1/\sqrt{1-x^2}, T_0(x) = 1, T_1(x) = x$$

Recurrence relation:

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad n \geq 1.$$

$$\text{Also, } T_n(x) = \cos(n\theta), \quad x = \cos \theta.$$

(c) *Hermite* Polynomials $H_n(x)$

$$a = -\infty, b = \infty, \rho(x) = e^{-x^2}, H_0(x) = 1, H_1(x) = 2x$$

Recurrence relation:

$$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x), \quad n \geq 1.$$

(d) *Laguerre* Polynomials $L_n(x)$

$$a = 0, b = \infty, \rho(x) = e^{-x}, L_0(x) = 1, L_1(x) = x - 1$$

Recurrence relation:

$$L_{n+1}(x) = (-2n - 1 + x)L_n(x) - n^2L_{n-1}(x), \quad n \geq 1.$$

□

Example 4.8

Find the least squares cubic approximation to $f(x) = \sin \pi x/2$ on $[-1, 1]$ with respect to the inner product

$$(g, h) = \int_{-1}^1 g(x)h(x)dx.$$

Solution: The Legendre polynomials are orthogonal on $[-1, 1]$ with this weight function. We have

$$P_0(x) = 1, \quad P_1(x) = x, \quad P_2(x) = \frac{1}{2}(3x^2 - 1), \quad P_3(x) = \frac{1}{2}(5x^3 - 3x),$$

$$(f, P_0) = 0, \quad (f, P_1) = 0.81057, \quad (f, P_2) = 0, \quad (f, P_3) = -0.06425,$$

$$(P_0, P_0) = 2, \quad (P_1, P_1) = 0.66667, \quad (P_2, P_2) = 0.20000, \quad (P_3, P_3) = 0.28571.$$

Then,

$$w(x) = \sum_{j=0}^3 \alpha_j P_j(x), \quad \alpha_j = \frac{(P_j, f)}{(P_j, P_j)},$$

so

$$w(x) = 1.5532x - 0.56223x^3 \approx \sin \frac{\pi x}{2}.$$

We obtain the results in Table 4.1

□

TABLE 4.1: Legendre polynomial approximation of degree 3 to $\sin(\pi x/2)$

x	$\sin(\pi x/2)$	$w(x)$
-1	-1.0000	-0.9909
-1/2	-0.7071	-0.7063
0	0.0000	0.0000
1/2	0.7071	0.7063
1	1.0000	0.9909

We have the following interesting and useful results about orthogonal polynomials.

PROPOSITION 4.2

If $\{\varphi_n(x)\}_{n=0}^{\infty}$ is an orthogonal sequence of polynomials and $\varphi_n(x)$ is of degree n for each n , then

$$\text{span}(1, x, x^2, \dots, x^m) = \text{span}(\varphi_0(x), \varphi_1(x), \dots, \varphi_m(x)) = P^m$$

for each m .

PROOF (by induction): First, $P^0 = \text{span}(1) = \text{span}(\varphi_0(x))$. Now suppose that

$$P^{m-1} = \text{span}(1, x, \dots, x^{m-1}) = \text{span}(\varphi_0(x), \dots, \varphi_{m-1}(x)),$$

and consider $\varphi_m(x)$:

$$\varphi_m(x) = c_m x^m + \sum_{i=0}^{m-1} c_i x^i = c_m x^m + \sum_{i=0}^{m-1} \hat{c}_i \varphi_i(x).$$

Thus,

$$x^m = \frac{1}{c_m} \varphi_m(x) - \sum_{i=0}^{m-1} \frac{\hat{c}_i}{c_m} \varphi_i(x).$$

Hence,

$$\text{span}(1, x, \dots, x^m) \subset \text{span}(\varphi_0(x), \varphi_1(x), \dots, \varphi_m(x)) \subset \text{span}(1, x, \dots, x^m).$$

Thus, $P^m = \text{span}(1, x, \dots, x^m) = \text{span}(\varphi_0(x), \varphi_1(x), \dots, \varphi_m(x))$ for each m . $\square$

REMARK 4.21 If $p \in P^n$, then $p(x) = \sum_{j=0}^n c_j \varphi_j(x) = \sum_{j=0}^n \frac{(p, \varphi_j)}{(\varphi_j, \varphi_j)} \varphi_j(x)$. $\square$

PROPOSITION 4.3

Let $\{\varphi_n(x)\}_{n=0}^{\infty}$ be an orthogonal family of polynomials on $C[a, b]$ with respect to the inner product $(f, g) = \int_a^b f(x)g(x)\rho(x)dx$, with $\rho(x) > 0$ for $a < x < b$. Let $\varphi_n(x)$ be of degree n . Then $\varphi_n(x)$ has n distinct zeros in open interval (a, b) .

PROOF (by contradiction) Let $x_1, x_2, \dots, x_m$ be zeros of $\varphi_n(x)$ for which

- (a) $a < x_i < b$ and
- (b) $\varphi_n(x)$ changes sign at x_i .

Since the degree of $\varphi_n(x)$ is n , we know that $m \leq n$. Assume $m < n$. (We will show that this is impossible.) By the definition of $x_1, x_2, \dots, x_m$, let

$$B(x) = \prod_{i=1}^m (x - x_i).$$

Then $\varphi_n(x)B(x) = (x - x_1) \dots (x - x_m)\varphi_n(x)$ does not change sign on (a, b) . To see this, the assumptions on $\varphi_n(x)$ imply that

$$\varphi_n(x) = h(x)(x - x_1)^{r_1}(x - x_2)^{r_2} \dots (x - x_m)^{r_m},$$

such that each r_i odd and such that $h(x)$ does not change sign on (a, b) . Thus,

$$B(x)\varphi_n(x) = h(x)(x - x_1)^{r_1+1}(x - x_2)^{r_2+1} \dots (x - x_m)^{r_m+1}$$

does not change sign on (a, b) . Consequently,

$$\int_a^b B(x)\varphi_n(x)\rho(x)dx \neq 0.$$

But since the degree of $B = m < n$,

$$B(x) = \sum_{j=1}^m c_j \varphi_j(x),$$

where $c_j = (B, \varphi_j)/(\varphi_j, \varphi_j)$ by Remark 4.21. Thus,

$$\int_a^b B(x)\varphi_n(x)\rho(x)dx = \sum_{j=1}^m c_j \int_a^b \varphi_j(x)\varphi_n(x)\rho(x)dx = 0, \text{ a contradiction.}$$

Thus, $m = n$ and the theorem follows, since $\varphi_n(x)$ can have at most n roots and the assumptions on $x_1, x_2, \dots, x_n$ imply that they must be distinct. $\square$

Example 4.9

We give roots of the Legendre and Chebyshev polynomials.

(a) Legendre polynomials (The interval is $(-1, 1)$.)

$$\begin{aligned} P_0(x) &= 1; \\ P_1(x) &= x, \quad x_1 = 0; \\ P_2(x) &= \frac{1}{2}(3x^2 - 1), \quad x_1 = -\frac{1}{\sqrt{3}}, \quad x_2 = \frac{1}{\sqrt{3}}; \\ P_3(x) &= \frac{1}{2}(5x^3 - 3x), \quad x_1 = 0, \quad x_2 = -\sqrt{3/5}, \quad x_3 = \sqrt{3/5}; \\ &\vdots \end{aligned}$$

(b) Chebyshev polynomials (The interval is $(-1, 1)$.)

$$T_n(x) = 0 \text{ at } x_i = \cos\left(\frac{2i+1}{n}\frac{\pi}{2}\right) \text{ for } i = 0, 1, \dots, n-1.$$

□

The following result tells us that any set of orthogonal polynomials is characterized by a three-term recurrence relation.

PROPOSITION 4.4

Let $\varphi_0(x) = 1$ and let

$$\varphi_1(x) = x - B_1, \text{ with } B_1 = \frac{(x\varphi_0, \varphi_0)}{(\varphi_0, \varphi_0)}, \text{ where } (f, g) = \int_a^b \rho(x)f(x)g(x)dx.$$

Suppose that $\varphi_k(x)$ is of degree k with leading term x^k for $k \geq 0$. Then $\{\varphi_k(x)\}_{k=0}^\infty$ satisfies the three-term recurrence

$$\varphi_k(x) = (x - B_k)\varphi_{k-1}(x) - C_k\varphi_{k-2}(x), \quad (4.23)$$

where

$$B_k = \frac{(x\varphi_{k-1}, \varphi_{k-1})}{(\varphi_{k-1}, \varphi_{k-1})} \quad \text{and} \quad C_k = \frac{(x\varphi_{k-1}, \varphi_{k-2})}{(\varphi_{k-2}, \varphi_{k-2})} \quad (4.24)$$

for $k \geq 2$ if and only if $\{\varphi_k\}_{k=0}^n$ are orthogonal on the interval $[a, b]$ with respect to the weight function $\rho(x)$.

PROOF We first suppose that the set $\{\varphi_k(x)\}_{k=0}^\infty$ is orthogonal. We need to show that $\varphi_k(x)$ satisfies the three term recurrence (4.23). First, we observe that $\varphi_{k+1}(x) - x\varphi_k(x)$ is of degree k . Then,

$$\varphi_{k+1}(x) - x\varphi_k(x) = -\hat{B}_{k+1}\varphi_k(x) - \hat{C}_{k+1}\varphi_{k-1}(x) - \dots - \hat{S}_{k+1}\varphi_0(x)$$

because

$$\varphi_{k+1}(x) - x\varphi_k(x) \in P^k = \text{span}(\varphi_0, \varphi_1, \dots, \varphi_k)$$

by Proposition 4.2. But

$$(\varphi_{k+1} - x\varphi_k, \varphi_j) = (\varphi_{k+1}, \varphi_j) - (\varphi_k, x\varphi_j) = 0$$

for $j = 0, 1, 2, \dots, k-2$. Thus, $\varphi_{k+1} - x\varphi_k = -\hat{B}_{k+1}\varphi_k - \hat{C}_{k+1}\varphi_{k-1}$, that is,

$$\varphi_{k+1} = (x - \hat{B}_{k+1})\varphi_k - \hat{C}_{k+1}\varphi_{k-1}.$$

Since $(\varphi_{k+1}, \varphi_k) = 0 = (x\varphi_k, \varphi_k) - \hat{B}_{k+1}(\varphi_k, \varphi_k)$, $\hat{B}_{k+1} = B_{k+1}$. Since $(\varphi_{k+1}, \varphi_{k-1}) = 0$, $\hat{C}_{k+1} = (x\varphi_k, \varphi_{k-1})/(\varphi_{k-1}, \varphi_{k-1}) = C_{k+1}$.

For the converse, we suppose that $\varphi_k(x)$ satisfies the three term recurrence (4.23). We need to show that $\{\varphi_k(x)\}_{k=0}^\infty$ are orthogonal. Since each $\varphi_k(x)$ is of the form $x^k + \{\text{lower-order terms}\}$, all denominators for B_k and C_k in (4.24) are nonzero. We will show by induction on k that $\int_a^b \rho(x)\varphi_k(x)\varphi_i(x)dx = 0$ for $i < k$. For $k = 1$,

$$(\varphi_1, \varphi_0) = (x\varphi_0, \varphi_0) - B_1(\varphi_0, \varphi_0) = 0.$$

Assume now that the result is true for $k = n-1$. Then,

$$\begin{aligned} (\varphi_n, \varphi_{n-1}) &= ((x - B_n)\varphi_{n-1}, \varphi_{n-1}) - C_n(\varphi_{n-2}, \varphi_{n-1}) \\ &= (x\varphi_{n-1}, \varphi_{n-1}) - B_n(\varphi_{n-1}, \varphi_{n-1}) - C_n(\varphi_{n-2}, \varphi_{n-1}) \\ &= 0 \quad (\text{by the definition of } B_n). \\ (\varphi_n, \varphi_{n-2}) &= ((x - B_n)\varphi_{n-1}, \varphi_{n-2}) - C_n(\varphi_{n-2}, \varphi_{n-2}) \\ &= (x\varphi_{n-1}, \varphi_{n-2}) - C_n(\varphi_{n-2}, \varphi_{n-2}) = 0 \quad (\text{by the definition of } C_n). \end{aligned}$$

For $i < n-2$,

$$\begin{aligned} (\varphi_n, \varphi_i) &= ((x - B_n)\varphi_{n-1}, \varphi_i) - C_n(\varphi_{n-2}, \varphi_i) = (x\varphi_{n-1}, \varphi_i) \\ &= (\varphi_{n-1}, x\varphi_i) = (\varphi_{n-1}, \varphi_{i+1} + B_{i+1}\varphi_i + C_{i+1}\varphi_{i-1}) = 0. \end{aligned}$$

□

4.3.7 Error in Least Squares Polynomial Approximation

We now consider the error in the least squares approximation to a function f in $C[-1, 1]$. Our first step along these lines is the following observation.

REMARK 4.22 Necessarily $\|f - p_n\|_2 \rightarrow 0$ as $n \rightarrow \infty$ in the L^2 -norm, where p_n is the least squares approximation of degree n to $f \in C[-1, 1]$. This follows from the Weierstrass Approximation Theorem since given $\epsilon > 0$ there exists a polynomial p such that $\|f - p\|_\infty < \epsilon/\sqrt{2}$. Hence, provided that n is sufficiently large,

$$\begin{aligned} \|f - p_n\|_2 &\leq \|f - p\|_2 = \left(\int_{-1}^1 (f(x) - p(x))^2 dx \right)^{1/2} \\ &\leq \|f - p\|_\infty \sqrt{2} < \epsilon. \end{aligned}$$

□

Now let's consider $\|f - p_n\|_\infty$. We have:

THEOREM 4.9

Suppose that $f \in C[-1, 1]$ and $p_n(x)$ is the least-squares approximation to $f(x)$ with respect to weight function $\rho(x) = (1 - x^2)^{-\frac{1}{2}}$, i.e., $p_n(x)$ is a linear combination of orthogonal Chebyshev polynomials. Then

$$\|f - p_n\|_\infty \leq \left(4 + \frac{4}{\pi^2} \ln n\right) E_n(f), \quad \text{where} \quad E_n(f) = \|f - p_n^*\|_\infty$$

and p_n^* is the best uniform approximation to f , i.e.,

$$\|f - p_n^*\|_\infty \leq \|f - u_n\|_\infty \quad \text{for any } u_n \in P^n.$$

PROOF See [75].

□

REMARK 4.23 If f satisfies a Lipschitz condition

$$|f(x) - f(z)| \leq k|x - z|$$

for $x, z \in [-1, 1]$, then it can be shown that

$$E_n(f) \leq 6k/n,$$

so $\|f - p_n\|_\infty \rightarrow 0$ as $n \rightarrow \infty$. Furthermore, if $f \in C^\ell[-1, 1]$, then

$$E_n(f) \leq \frac{c}{n^\ell}$$

for some constant c independent of n . Thus, if f is smooth, the least squares approximation rapidly improves as n increases. In particular, if $f \in C^\infty[-1, 1]$, the error in the Chebyshev least squares approximation goes to zero more rapidly than any finite power of $1/n$ as $n \rightarrow \infty$. □

4.3.8 Minimax (Best Uniform) Approximation

We now briefly study the best uniform approximation. (See [75] for a thorough treatment of best uniform approximations.)

DEFINITION 4.9 Let $V = C[a, b]$ be a normed vector space with norm

$$\|v\|_\infty = \max_{a \leq x \leq b} |v(x)|.$$

Let $W = P^n \subset V$ be the set of polynomials of degree n or less. The best uniform approximation $p_n^*(x)$ to $f(x) \in C[a, b]$ (existence is guaranteed by Theorem 4.1) satisfies $\|p_n^* - f\|_\infty \leq \|p_n - f\|_\infty$ for all $p_n \in P^n$, and is called the minimax or best uniform approximation to f . It is called “minimax” because

$$\|p_n^* - f\|_\infty = \min_{p_n \in P^n} \left\{ \max_{a \leq x \leq b} |p_n(x) - f(x)| \right\}.$$

DEFINITION 4.10 Let $f(x)$ be defined on $[a, b]$; the modulus of continuity of $f(x)$ on $[a, b]$, $\omega(\delta)$ is defined for $\delta > 0$ by

$$\omega(\delta) = \sup_{\substack{x_1, x_2 \in [a, b] \\ |x_1 - x_2| \leq \delta}} |f(x_1) - f(x_2)|.$$

REMARK 4.24 ω depends on f , $[a, b]$, and δ , so $\omega(\delta)$ is actually shorthand for $\omega(f; [a, b]; \delta)$. Notice that if f is Lipschitz continuous on $[a, b]$, then $\omega(\delta) \leq L\delta$. $\square$

Several properties of modulus of continuity follow from Definition 4.10.

LEMMA 4.1

If $0 \leq \delta_1 \leq \delta_2$, then $\omega(\delta_1) \leq \omega(\delta_2)$.

LEMMA 4.2

$f \in C[a, b]$ if and only if $\lim_{\delta \rightarrow 0} \omega(\delta) = 0$. (f is uniformly continuous on $[a, b]$ if $f \in C[a, b]$.)

LEMMA 4.3

If $\lambda > 0$, then $\omega(\lambda\delta) \leq (1 + \lambda)\omega(\delta)$.

PROOF Let n be an integer such that $n \leq \lambda < n + 1$, so that $\omega(\lambda\delta) \leq \omega((n + 1)\delta)$. Suppose that $|x_1 - x_2| = (n + 1)\delta$ and say $x_2 > x_1$. With the points $z_j = x_1 + j(x_2 - x_1)/(n + 1)$, $j = 0, 1, \dots, n + 1$, divide $[x_1, x_2]$ into $n + 1$ equal parts each of length $(x_2 - x_1)/(n + 1)$. Then

$$|f(x_2) - f(x_1)| = \left| \sum_{j=0}^n f(z_{j+1}) - f(z_j) \right| \leq \sum_{j=0}^n |f(z_{j+1}) - f(z_j)| \leq (n + 1)\omega(\delta).$$

Thus, $\omega((n + 1)\delta) \leq (n + 1)\omega(\delta)$. This and $(n + 1) \leq \lambda + 1$ thus imply

$$\omega(\lambda\delta) \leq \omega((n + 1)\delta) \leq (\lambda + 1)\omega(\delta).$$

4.3.8.1 Error in the Best Uniform Approximation

We now consider the error in the best uniform approximation.

DEFINITION 4.11 We define

$$E_n(f; [a, b]) = E_n(f) = \|f - p_n^*\|_\infty$$

to be the error in the best uniform approximation.

THEOREM 4.10

$$E_n(f; [0, 1]) = \|f - p_n^*\|_\infty \leq \frac{3}{2} \omega\left(\frac{1}{\sqrt{n}}\right) \text{ for } f \in C[0, 1].$$

PROOF Recall that

$$B_n(f; x) = \sum_{k=0}^n f\left(\frac{k}{n}\right) \binom{n}{k} x^k (1-x)^{n-k}$$

is the n -th Bernstein polynomial of f . Thus, $\|f - p_n^*\|_\infty \leq \|f - B_n(f; x)\|_\infty$. But

$$\begin{aligned} |f(x) - B_n(f; x)| &= \left| \sum_{k=0}^n \left[f(x) - f\left(\frac{k}{n}\right) \right] \binom{n}{k} x^k (1-x)^{n-k} \right| \\ &\leq \sum_{k=0}^n \left| f(x) - f\left(\frac{k}{n}\right) \right| \binom{n}{k} x^k (1-x)^{n-k} \\ &\leq \sum_{k=0}^n \omega\left(\left|x - \frac{k}{n}\right|\right) \binom{n}{k} x^k (1-x)^{n-k} \end{aligned}$$

Applying Lemma 4.3 gives

$$\begin{aligned} \omega\left(\left|x - \frac{k}{n}\right|\right) &= \omega\left(n^{1/2} \left|x - \frac{k}{n}\right| n^{-1/2}\right) \\ &\leq \left(1 + n^{1/2} \left|x - \frac{k}{n}\right|\right) \omega\left(n^{-1/2}\right). \end{aligned}$$

Thus,

$$\begin{aligned} |f(x) - B_n(f; x)| &\leq \left[\sum_{k=0}^n \left(1 + n^{1/2} \left|x - \frac{k}{n}\right|\right) \binom{n}{k} x^k (1-x)^{n-k} \right] \omega(n^{-1/2}) \\ &\leq \omega\left(n^{-1/2}\right) \left[1 + n^{1/2} \sum_{k=0}^n \left|x - \frac{k}{n}\right| \binom{n}{k} x^k (1-x)^{n-k} \right]. \end{aligned}$$

By the Cauchy–Schwarz inequality,

$$\begin{aligned}
 & \sum_{k=0}^n \left| x - \frac{k}{n} \right| \sqrt{\binom{n}{k} x^k (1-x)^{n-k}} \sqrt{\binom{n}{k} x^k (1-x)^{n-k}} \\
 & \leq \left(\sum_{k=0}^n \left(x - \frac{k}{n} \right)^2 \binom{n}{k} x^k (1-x)^{n-k} \right)^{\frac{1}{2}} \left(\sum_{k=0}^n \binom{n}{k} x^k (1-x)^{n-k} \right)^{\frac{1}{2}} \\
 & = \left(\frac{x(1-x)}{n} \right)^{1/2} \\
 & \leq \frac{1}{2n^{1/2}}.
 \end{aligned}$$

(See the proof of Weierstrass approximation theorem, starting on page 205. There, we saw that $B_n(1; x) = 1$, $B_n(x; x) = x$ and $B_n(x^2; x) = x^2 + x(1-x)/n$.) Hence,

$$|f(x) - B_n(f; x)| \leq \omega \left(n^{-1/2} \right) \left[1 + \frac{n^{1/2}}{2n^{1/2}} \right] = \frac{3}{2} \omega \left(n^{-1/2} \right).$$

Therefore,

$$\|f - p_n^*\|_\infty \leq \frac{3}{2} \omega \left(\frac{1}{\sqrt{n}} \right).$$

□

REMARK 4.25 Better error estimates than given in Theorem 4.10 can be obtained. In particular, Jackson's Theorem [75] gives:

$$(a) \text{ if } f \in C[-1, 1], \text{ then } E_n(f; [-1, 1]) \leq 6\omega \left(\frac{1}{n} \right);$$

$$(b) \text{ if } f \in C[a, b], \text{ then } E_n(f; [a, b]) \leq 6\omega \left(\frac{b-a}{2n} \right) \text{ (by (a) and a change of variables).}$$

□

REMARK 4.26 Suppose that $f \in C^1[-1, 1]$ and $|f'(x)| \leq M$ on $[-1, 1]$. Then $E_n(f; [-1, 1]) \leq 6M/n$. □

PROOF (of Remark 4.26)

$$\omega \left(\frac{1}{n} \right) = \sup_{\substack{x_1, x_2 \in [-1, 1] \\ |x_1 - x_2| \leq 1/n}} |f(x_1) - f(x_2)| = \sup_{\substack{x_1, x_2 \in [-1, 1] \\ |x_1 - x_2| \leq 1/n}} |f'(\xi(x_1, x_2))| |x_1 - x_2| \leq \frac{M}{n},$$

where $\xi(x_1, x_2) \in [x_1, x_2]$ is the point guaranteed by the Intermediate Value Theorem to have the property $f(x_1) - f(x_2) = f'(\xi(x_1, x_2))(x_1 - x_2)$. Remark 4.26 then follows from Remark 4.25. $\square$

REMARK 4.27 If $f \in C^k[-1, 1]$, $n > k$, and $|f^{(k)}(x)| \leq M_k$ on $[-1, 1]$, then it can be shown that

$$E_n(f; [-1, 1]) \leq \frac{c_k}{n^k} M_k, \quad \text{where } c_k \text{ is independent of } n.$$

$\square$

4.3.8.2 Characterization of the Best Uniform Approximation

Let $e(x) = f(x) - p_n^*(x)$, where $p_n^*(x)$ is the best uniform approximation to $f(x)$ in P^n . Thus, $\|e\|_\infty = E_n(f; [a, b])$.

THEOREM 4.11

There exists (at least) two distinct points $x_1, x_2 \in [a, b]$ such that

$$|e(x_1)| = |e(x_2)| = E_n(f; [a, b]) \quad \text{and} \quad e(x_1) = -e(x_2).$$

PROOF The continuous curve $y = e(x)$ is constrained to lie between $y = \pm E_n(f)$ for $a \leq x \leq b$ and touches at least one of these lines. We wish to prove that it must touch both. Suppose it doesn't touch both. Then, as is illustrated in Figure 4.5, a better approximation to f than p_n^* exists. Assume

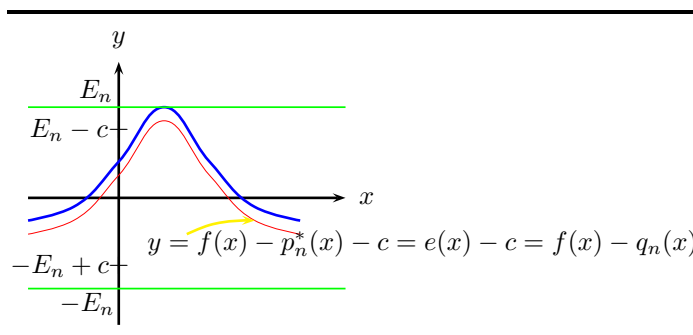


FIGURE 4.5: A better approximation to f than p_n^* .

that $e(x) > -E_n(f)$ for all $x \in [a, b]$. Then

$$\min_{a \leq x \leq b} e(x) = m > -E_n(f)$$

and $c = (E_n(f) + m)/2 > 0$. Since $q_n(x) = p_n^*(x) + c \in P^n$, it follows that $f(x) - q_n(x) = e(x) - c$ and

$$-(E_n(f) - c) = m - c \leq e(x) - c \leq E_n(f) - c.$$

We thus have $\|f - q_n\|_\infty = E_n(f) - c$, contradicting the definition of $E_n(f)$. Hence, there must be a point x_1 such that $e(x_1) = -E_n(f)$. Similarly, there is a point x_2 such that $e(x_2) = E_n(f)$. $\square$

COROLLARY 4.1

The best uniform approximating constant to $f(x)$ is

$$p_0^* = \frac{1}{2} \left[\max_{a \leq x \leq b} f(x) + \min_{a \leq x \leq b} f(x) \right]$$

and

$$E_0(f) = \frac{1}{2} \left[\max_{a \leq x \leq b} f(x) - \min_{a \leq x \leq b} f(x) \right].$$

PROOF If d is any other constant than p_0^* , then $e(x) = f(x) - d$ does not satisfy Theorem (4.11). $\square$

The argument in Theorem (4.11) can be extended as follows.

THEOREM 4.12

Suppose that $f \in C[a, b]$, p_n^* is the best uniform approximation on $[a, b]$ to f from P^n if and only if there exists an alternating set consisting of $n + 2$ points for the error $e(x) = f(x) - p_n^*(x)$, i.e., there exists

$$a \leq x_0 < x_1 < x_2 < \cdots < x_{n+1} \leq b$$

such that

$$|e(x_j)| = \|f - p_n^*\|_\infty$$

for $j = 0, 1, 2, \dots, n + 1$ and

$$e(x_j) = -e(x_{j+1})$$

for $j = 0, 1, 2, \dots, n$. We see this graphically in Figure 4.6. (It can be shown that this theorem implies that p_n^* is the unique best approximation to f from P^n ; see [75].)

The alternating error given in Theorem 4.12 is often called the *minimax equi-oscillation property*, and a variants of Theorem 4.12 are called *Chebyshev's Equi-Oscillation Theorem*. Note that Chebyshev polynomials have the equi-oscillation property on $[-1, 1]$.

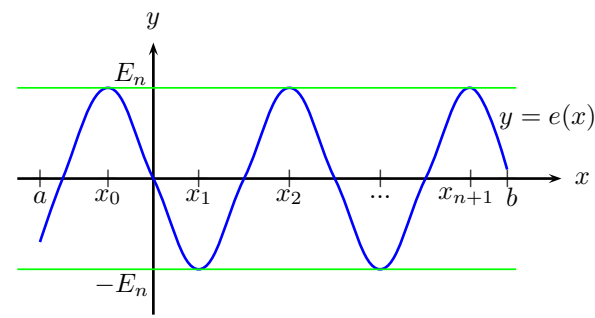


FIGURE 4.6: $n + 1$ alternations of the best approximation error (Theorem 4.12).

REMARK 4.28 In general, it is impossible to find in a finite number of steps the best uniform approximation $p_n^* \in P^n$ to a given function $f \in C[a, b]$ in the uniform norm. A strategy used is to replace $[a, b]$ by a finite set of distinct points in $[a, b]$ and solve the approximation problem on this finite point set. The *Exchange Method of Remez* or *Remez Algorithm* is a computational procedure for finding $p_n^*(x)$ on a finite point set. See, for example, [74]. $\square$

REMARK 4.29 Since minimax approximations are difficult to find, near minimax approximations are useful. Consider the following two near minimax approximations.

1. Let $f \in C[-1, 1]$. We saw that if

$$p_n(x) = \sum_{i=1}^n \frac{(f, T_i)}{(T_i, T_i)} T_i(x),$$

where $T_i(x)$ is the i -th Chebyshev polynomial and $p_n(x)$ is the least squares approximation to $f(x)$ with respect to weight function $\rho(x) = (1 - x^2)^{-\frac{1}{2}}$, then

$$\|f - p_n\|_\infty \leq \left[4 + \frac{4}{\pi^2} \ln n \right] E_n(f),$$

where $E_n(f) = \|f - p_n^*\|_\infty$ and p_n^* is the minimax approximation. Thus, $\|f - p_n\|_\infty$ decreases roughly in proportion to $E_n(f)$, and $p_n(x)$ is called a *near minimax approximation*. (Note that $g(t) \in C[a, b]$ can be converted by change of variables to $f(x) \in C[-1, 1]$.)

2. Consider the Lagrange interpolating polynomial $p_n(x)$ with interpolating nodes $x_i = \cos\left(\frac{2i+1}{n+1} \cdot \frac{\pi}{2}\right)$, $0 \leq i \leq n$, on $[-1, 1]$. We saw that

$$\|f - p_n\|_\infty \leq \frac{1}{(n+1)!} \|f^{(n+1)}\|_\infty 2^{-n}.$$

This interpolating polynomial is sometimes considered a minimax approximation.

□

4.3.9 Interval (Rigorous) Bounds on the Errors

Whether we are considering Taylor polynomial approximation, interpolation, least squares, or minimax approximation, the error term in the approximation can be put in the general form

$$f(x) = p(x) + K(x)M(f;x) \quad \text{for } x \in [a, b]. \tag{4.25}$$

We list K and M for various approximations⁸ in Table 4.2. In such cases,

TABLE 4.2: Error factors K and M in polynomial approximations $f(x) = p(x) + K(x)M(f;x)$.

Type of approximation	K	$M(f)$
degree n Taylor polynomial	$\frac{1}{(n+1)!}(x-x_0)^{n+1}$	$f^{(n+1)}(\xi(x)), \xi \in [a, b]$ unknown
polynomial interpolation at $n+1$ points	$\frac{1}{(n+1)!} \prod_{i=0}^n (x-x_i)$	$f^{(n+1)}(\xi(x)), \xi \in [a, b]$ unknown
least squares approximation of degree n in the Chebyshev norm, $f \in C^0[a, b]$	$\left\{ 4 + \frac{4}{\pi^2} \ln(n) \right\}$	$6\omega\left(f; [a, b]; \frac{b-a}{2n}\right)$

p and K can be evaluated explicitly, while $M(f;x)$ can be estimated using interval arithmetic. We illustrated how to do this for $f(x) = e^x$, using a degree-5 Taylor polynomial, in Example 1.17 on page 28. We elaborate here: In addition to bounding particular values of the function, a maximum error of approximation and rigorous bounds valid for all of $[a, b]$ can be inferred. In particular, the polynomial part $p(x)$ is evaluated at a point (but using outwardly rounded interval arithmetic to maintain mathematical rigor), and the error part is evaluated with interval arithmetic.

⁸The error of approximation of smooth functions by Chebyshev polynomials can be much less than for nonsmooth (merely C^0) functions, as is indicated in Remark 4.27 combined with Theorem 4.9; however, bounds on the error may be more complicated to find in this case.

Example 4.10

Consider approximating $\sin(x)$, $x \in [-0.1, 0.1]$ by a degree-5

1. Taylor polynomial about zero,
2. interpolating polynomial at the points $x_k = -.1 + .04k$, $0 \leq k \leq 5$.

For the Taylor polynomial, we observe that the fifth degree Taylor polynomial is the same as the sixth degree Taylor polynomial, and we have

$$\sin(x) \in x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \frac{1}{5040}x^7 \sin(\xi) \quad \text{for some } \xi \in [-0.1, 0.1]. \quad (4.26)$$

We can replace $\sin(\xi)$ by an appropriate interval to get a pointwise estimate; for example,

$$\begin{aligned} \sin(0.05) &\in .05 - \frac{.05^3}{6} + \frac{.05^5}{120} - \frac{.05^7}{5040} [0, 0.05] \\ &\subseteq [\underline{0.049979169270821}, \underline{0.04997916927084}], \end{aligned}$$

where the above bounds are mathematically rigorous. Here, K was evaluated at the point x , but, $\sin(\xi)$ was replaced by $\sin([0, 0.05])$. Similarly,

$$\begin{aligned} \sin(-0.01) &\in (-.01) - \frac{(-.01)^3}{6} + \frac{(-.01)^5}{120} - \frac{(-.01)^7}{5040} [-0.01, 0] \\ &\subseteq [-\underline{0.00999983333417}, -\underline{0.00999983333416}]. \end{aligned}$$

Thus, since we know $\sin(x)$ is monotonic for $x \in [-0.01, 0.05]$, $[-\underline{0.00999983333417}, \underline{0.04997916927084}]$ represents a fairly sharp bound on the range $\{\sin(x) \mid x \in [-0.01, 0.05]\}$. Alternately, it may be more convenient in some contexts to evaluate K and M over the entire interval, although this leads to a less sharp result. Using that technique, we would have

$$\begin{aligned} \sin(0.05) &\in .05 - \frac{.05^3}{6} + \frac{.05^5}{120} + \frac{[-0.1, 0.1]^7}{5040} [-0.1, 0.1] \\ &\subseteq .05 - \frac{.05^3}{6} + \frac{.05^5}{120} - \\ &\quad [-0.19841269841270 \times 10^{-11}, 0.19841269841270 \times 10^{-11}] \\ &\subseteq [\underline{0.04997916926884}, \underline{0.04997916927282}], \end{aligned}$$

and

$$\begin{aligned} \sin(-0.01) &\in (-.01) - \frac{(-.01)^3}{6} + \frac{(-.01)^5}{120} - \frac{[-0.1, 0.1]^7}{5040} [-0.1, 0.1] \\ &\subseteq (-.01) - \frac{(-.01)^3}{6} + \frac{(-.01)^5}{120} - \\ &\quad [-0.19841269841270 \times 10^{-11}, 0.19841269841270 \times 10^{-11}] \\ &\subseteq [-\underline{0.00999983333616}, -\underline{0.00999983333218}], \end{aligned}$$

thus obtaining (somewhat less sharp) bounds

$$[-0.00999983333616, 0.04997916927282]$$

on the range $\{\sin(x) \mid x \in [-0.01, 0.05]\}$.

In general, substituting intervals into the polynomial approximation itself does not give sharp bounds on the range. For example,

$$\begin{aligned} \sin([-0.01, 0.05]) &\in ([-.01, .05]) - \frac{([-0.01, .05])^3}{6} + \frac{([-0.01, .05])^5}{120} - \\ &\quad [-0.19841269841270 \times 10^{-11}, 0.19841269841270 \times 10^{-11}] \\ &\subseteq [-0.01002083333616, 0.05000016927282]. \end{aligned}$$

Nonetheless, in some contexts in which there is no alternative, this technique gives usable bounds.

Computing bounds based on the interpolating polynomial is similar to computing bounds based on the Taylor polynomial, and is left as Exercise 22. $\square$

Moduli of continuity, as well as Lipschitz constants, can be easily estimated when $f \in C^1[a, b]$. In particular, in such cases,

$$\omega(f; [a, b]; \delta) \leq \mathbf{f}'([a, b])\delta, \quad (4.27)$$

where $\mathbf{f}'([a, b])$ is an interval evaluation of the derivative f' over $[a, b]$ (or any other set of bounds on the range of $\mathbf{f}$ over $[a, b]$).

We now return to the concept, analogous to composite integration, of dividing the interval of approximation into subintervals, and using a different polynomial over each subinterval.

4.4 Piecewise Polynomial Approximation

Piecewise polynomials are commonly used approximations. They are easy to work with, they can provide good approximations, and they are widely used in computer graphics. In addition, piecewise polynomials are employed, for example, in finite element methods. Good references for piecewise polynomial approximation are [52] and [80].

A simple type of piecewise polynomial is approximation by a line segment in each subinterval.

4.4.1 Piecewise Linear Interpolation

DEFINITION 4.12 *Given a partition*

$$\Delta : a = x_0 < x_1 < \cdots < x_{n-1} < x_n = b$$

of $[a, b]$, the set L_Δ of all continuous piecewise linear polynomials on $[a, b]$ with respect to Δ is

$$L_\Delta = \{\varphi(x) \in C[a, b] : \varphi(x) \text{ is linear on each } [x_i, x_{i+1}], 0 \leq i \leq N-1 \text{ of } \Delta.\}$$

Graphically, $\varphi \in L_\Delta$ may appear, for example, as in Figure 4.7.

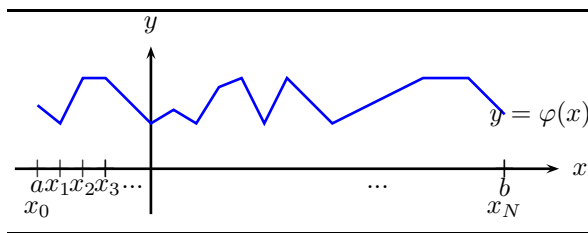


FIGURE 4.7: An example of a piecewise linear function.

PROPOSITION 4.5

L_Δ is an $(N+1)$ -dimensional subspace of $C[a, b]$.

PROOF L_Δ is a subspace of $C[a, b]$, since L_Δ is closed under addition and scalar multiplication and $0 \in L_\Delta$. To complete the proof, we find a basis of $N+1$ functions. This basis for L_Δ consists of the well-known *hat functions* $\varphi_i(x)$, $0 \leq i \leq N$, defined by

$$\begin{aligned} \varphi_0(x) &= \begin{cases} \frac{x_1 - x}{x_1 - x_0}, & x_0 \leq x \leq x_1, \\ 0, & \text{otherwise,} \end{cases} \\ \varphi_N(x) &= \begin{cases} \frac{x - x_{N-1}}{x_N - x_{N-1}}, & x_{N-1} \leq x \leq x_N, \\ 0, & \text{otherwise,} \end{cases} \\ \varphi_i(x) &= \begin{cases} \frac{x - x_{i-1}}{x_i - x_{i-1}}, & x_{i-1} \leq x \leq x_i, \\ \frac{x_{i+1} - x}{x_{i+1} - x_i}, & x_i \leq x \leq x_{i+1}, \end{cases} \quad \text{for } 1 \leq i \leq N. \end{aligned}$$

These hat functions are depicted graphically in Figure 4.8. Notice that

$$\begin{cases} \varphi_i \in L_\Delta \text{ for } i = 0, 1, 2, \dots, N, \\ \varphi_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \end{cases}$$

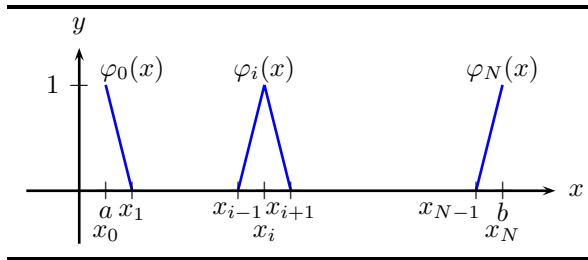


FIGURE 4.8: Graphs of the “hat” functions $\varphi_i(x)$.

We will show that $\{\varphi_i\}_{i=0}^N$ spans L_Δ , that $\{\varphi_i\}_{i=0}^N$ is a linearly independent set, and thus that $\{\varphi_i\}_{i=0}^N$ forms a basis for L_Δ .

Linear Independence: Let $\sum_{j=0}^N c_j \varphi_j(x) = 0$ for every $x \in [a, b]$. Setting $x = x_i$, $0 \leq i \leq N$, we conclude that $c_i = 0$, $0 \leq i \leq N$. Thus, $\{\varphi_i\}_{i=0}^N$ is a linearly independent set. (Recall that $\{y_i(x)\}_{i=0}^n$ is a linearly dependent set on an interval if and only if there are constants $k_0, k_1, \dots, k_N$, not all zero, such that $\sum_{i=0}^N k_i y_i(x) = 0$ for all x on the interval.)

Spans L_Δ : Let $f(x) \in L_\Delta$ and let $f_i = f(x_i)$, $0 \leq i \leq N$. Then $f(x) = \sum_{j=0}^N f_j \varphi_j(x)$, since the right side coincides with $f(x)$ at $x = x_i$, $0 \leq i \leq N$, and is linear in each subinterval $[x_i, x_{i+1}]$. (This also shows that $\{\varphi_i\}_{i=0}^N$ is a collocating basis.) $\square$

PROPOSITION 4.6

Given $f \in C[a, b]$, there is a unique $\Phi \in L_\Delta$ which satisfies $f(x_i) = \Phi(x_i)$, $0 \leq i \leq N$.

PROOF Define $\Phi(x) = \sum_{j=0}^n f(x_j) \varphi_j(x)$, where the $\varphi_j(x)$ are the basis functions defined after the proof of Proposition 4.5. Clearly, $\Phi \in L_\Delta$ and $\Phi(x_i) = f(x_i)$, $0 \leq i \leq N$. Moreover, if two different such Φ 's existed, say Φ_1 and Φ_2 with $\Phi_1(x_j) = \Phi_2(x_j) = f(x_j)$, $0 \leq j \leq N$, then the piecewise linear function $\Phi_1 - \Phi_2$, being zero at x_j , $0 \leq j \leq N$, would have to be zero everywhere on $[a, b]$ (since the linear interpolant at x_j and x_{j+1} is unique). $\square$

DEFINITION 4.13 We call Φ , defined in Proposition (4.6), the L_Δ - interpolant of f , and denote it by $\Phi(x) = I_N f(x)$.

REMARK 4.30 $I_N f(x) = \sum_{j=0}^N f(x_j) \varphi_j(x)$ is easily obtained. $\square$

REMARK 4.31 $I_N : C[a, b] \rightarrow L_\Delta$ is a linear operator, i.e.,

$$I_N (af_1(x) + bf_2(x)) = aI_N f_1(x) + bI_N f_2(x).$$

$\square$

We now wish to estimate the error in piecewise linear interpolation, i.e.,

$$\|f - I_N f\|_\infty = \max_{x \in [a, b]} |f(x) - I_N f(x)|.$$

Notation 4.1 Let $h = \max_{0 \leq i \leq N-1} (x_{i+1} - x_i)$ and $D^n f(x) = f^{(n)}(x)$.

We have

THEOREM 4.13

Let $f \in C^2[a, b]$. Then $\|f - I_N f\|_\infty \leq \frac{1}{8} h^2 \|D^2 f\|_\infty$.

PROOF

$$\begin{aligned} \|f - I_N f\|_\infty &= \max_{0 \leq i \leq N-1} \left[\max_{x_i \leq x \leq x_{i+1}} |f(x) - I_N f(x)| \right] \\ &= \max_i \max_{x_i \leq x \leq x_{i+1}} \left| f(x) - \left(f(x_i) \frac{(x_{i+1} - x)}{h_i} + f(x_{i+1}) \frac{(x - x_i)}{h_i} \right) \right|, \end{aligned} \quad (4.28)$$

where $h_i = x_{i+1} - x_i$. By Taylor's Theorem,

$$\begin{aligned} f(x_i) &= f(x) + f'(x)(x_i - x) + \int_x^{x_i} (x_i - t) f''(t) dt \\ f(x_{i+1}) &= f(x) + f'(x)(x_{i+1} - x) + \int_x^{x_{i+1}} (x_{i+1} - t) f''(t) dt. \end{aligned}$$

Substituting these expressions into (4.28) gives

$$\begin{aligned} \|f - I_N f\|_\infty &= \max_i \max_{x_i \leq x \leq x_{i+1}} \left| \frac{x_{i+1} - x}{h_i} \int_x^{x_i} (x_i - t) f''(t) dt \right. \\ &\quad \left. + \frac{x_i - x}{h_i} \int_x^{x_{i+1}} (x_{i+1} - t) f''(t) dt \right|. \end{aligned}$$

Thus, by the triangle inequality and taking $\|D^2 f\|_\infty$ outside the integrals and

the maximum,

$$\begin{aligned}
 & \|f - I_N f\|_\infty \\
 & \leq \|D^2 f\|_\infty \max_i \max_{x_i \leq x \leq x_{i+1}} \cdot \\
 & \quad \cdot \left[\frac{x_{i+1} - x}{h_i} \int_{x_i}^x (t - x_i) dt + \frac{x - x_i}{h_i} \int_{x_{i+1}}^x (t - x_{i+1}) dt \right] \\
 & = \|D^2 f\|_\infty \max_i \max_{x_i \leq x \leq x_{i+1}} \left[\frac{x_{i+1} - x}{h_i} \frac{(x - x_i)^2}{2} + \frac{x - x_i}{h_i} \frac{(x_{i+1} - x)^2}{2} \right] \\
 & = \|D^2 f\|_\infty \max_i \max_{x_i \leq x \leq x_{i+1}} \left[\frac{1}{2} (x_{i+1} - x)(x - x_i) \right] \\
 & = \|D^2 f\|_\infty \max_{0 \leq i \leq N-1} \frac{1}{8} (x_{i+1} - x_i)^2 = \frac{h^2}{8} \|D^2 f\|_\infty.
 \end{aligned}$$

□

REMARK 4.32 For $f \in C^2[a, b]$, it can also be shown that

$$\|D(f - I_N f)\|_\infty \leq \frac{1}{2} h \|D^2 f\|_\infty.$$

(See [80].)

□

REMARK 4.33 For $f \in C^1[a, b]$, it can be shown that

$$\|f - I_N f\|_\infty \leq \frac{1}{2} h \|Df\|_\infty.$$

□

REMARK 4.34 For $f \in C[a, b]$, it is straightforward to show that

$$\|f - I_N f\|_\infty \rightarrow 0$$

as $h \rightarrow 0$.

□

Example 4.11

Consider $f(x) = \ln x$ on the interval $[2, 4]$. We want to find h that will guarantee that the piecewise linear interpolant of $f(x)$ on $[2, 4]$ has an error of at most 10^{-4} . We will assume that $|x_{i+1} - x_i| = h$ for $0 \leq i \leq N-1$. Then

$$\|f - I_N f\|_\infty \leq \frac{h^2}{8} \|f''\|_\infty = \frac{h^2}{8} \max_{2 \leq x \leq 4} \left| \frac{1}{x^2} \right| = \frac{h^2}{32} \leq 10^{-4}.$$

Thus $h^2 \leq 32 \times 10^{-4}$, $h \leq 0.056$, giving $N = (4 - 2)/h \geq 36$. $\square$

Although hat functions and piecewise linear functions are frequently used in practice, it is desirable in some applications, such as computer graphics, for the interpolant to be smoother (say C^1 , C^2 , or even higher) at the mesh points x_i of Δ . Special piecewise cubic polynomials, which we consider next, are commonly used for this purpose.

4.4.2 Cubic Spline Interpolation

DEFINITION 4.14 *The set of cubic splines on $[a, b]$ with respect to the partition Δ is defined as*

$$S_\Delta = \{\varphi(x) \in C^2[a, b] : \varphi(x) \text{ is a cubic polynomial on each subinterval } [x_j, x_{j+1}], 0 \leq j \leq N-1, \text{ of } \Delta\}.$$

REMARK 4.35 $\varphi \in S_\Delta$ has C^2 smoothness, i.e., $\varphi \in S_\Delta$ has two continuous derivatives at each point, including the mesh points. $\square$

We first develop a basis for S_Δ . For convenience, we assume here a uniform mesh, i.e., $x_{j+1} - x_j = h$ for all j , and let

$$s_j(x) = \begin{cases} 0, & x > x_{j+2} \\ \frac{1}{6h^3}(x_{j+2} - x)^3, & x_{j+1} \leq x \leq x_{j+2}, \\ \frac{1}{6} + \frac{1}{2h}(x_{j+1} - x) + \frac{1}{2h^2}(x_{j+1} - x)^2 \\ \quad - \frac{1}{2h^3}(x_{j+1} - x)^3, & x_j \leq x \leq x_{j+1}, \\ \frac{2}{3} - \frac{1}{h^2}(x - x_j)^2 - \frac{1}{2h^3}(x - x_j)^3, & x_{j-1} \leq x \leq x_j, \\ \frac{1}{6h^3}(x - x_{j-2})^3 & x_{j-2} \leq x \leq x_{j-1}, \\ 0, & x < x_{j-2}. \end{cases} \quad (4.29)$$

DEFINITION 4.15 *The function $s_j(x)$ defined by (4.29) is called a B-spline centered at $x = x_j$ with respect to the partition Δ with a uniform mesh.*

REMARK 4.36 It is straightforward to show that $s'_j(x)$ and $s''_j(x)$ are continuous, so $s_j(x) \in S_\Delta$ (Exercise 23 on page 287). $\square$

We now introduce two extra points $x_{-1} = x_0 - h$ and $x_{N+1} = x_N + h$ and also consider the B-splines $s_{-1}(x)$ and $s_{N+1}(x)$ centered at x_{-1} and x_{N+1} . Now let

$$\varphi_j(x) = \begin{cases} 0, & x < a, \\ s_j(x), & a \leq x \leq b, \\ 0, & x > b, \end{cases} \text{ for } -1 \leq j \leq N+1.$$

The φ_j 's are depicted graphically in Figure 4.9.

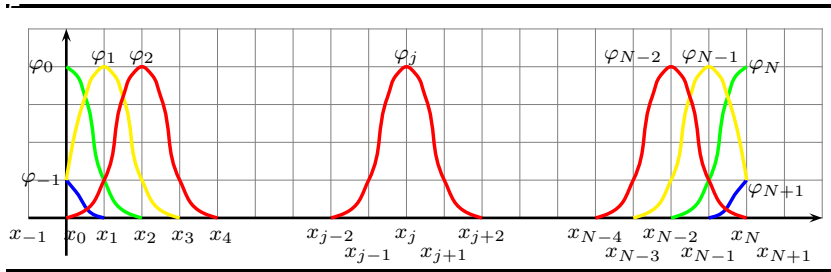


FIGURE 4.9: B-spline basis functions.

THEOREM 4.14

The functions $\{\varphi_j(x)\}_{j=-1}^{N+1}$ form a basis for S_Δ .

PROOF First we show linear independence, then we show that $\{\varphi_j(x)\}_{j=-1}^{N+1}$ spans S_Δ .

Proof of Linear Independence: Let $\sum_{j=-1}^{N+1} c_j \varphi_j(x) = 0$ for all x . In particular,

$$\sum_{j=-1}^{N+1} c_j \varphi_j(x_i) = 0, \quad 0 \leq i \leq N, \quad \text{and} \quad \sum_{j=-1}^{N+1} c_j \varphi'_j(x_k) = 0 \quad \text{for } k = 0 \text{ and } k = N.$$

Using the definition (4.29) of the B-spline s_j , we see for $c = [c_0, c_1, \dots, c_N]^T$ that $Ac = 0$, $c_{-1} = c_1$, and $c_{N+1} = c_{N-1}$, where

$$A = \begin{pmatrix} 4 & 2 & 0 & & \cdots & & 0 \\ 1 & 4 & 1 & & & & \vdots \\ 0 & 1 & 4 & 1 & & & \\ \vdots & & & & \ddots & & 0 \\ & & & & & 1 & 4 & 1 \\ 0 & & \cdots & & & 0 & 2 & 4 \end{pmatrix}. \quad (4.30)$$

Since $|a_{ii}| > \sum_{\substack{j=0 \\ j \neq i}}^N |a_{ij}|$ for $i = 0, 1, 2, \dots, N$, we see that A is strictly diagonally

dominant and hence nonsingular. Thus, $c = 0$, that is, $c_{-1} = c_0 = c_1 = \dots = c_{N+1} = 0$. Therefore, $\{\varphi_j\}_{j=-1}^{N+1}$ is a linearly independent set.

Proof that $\{\varphi_j(x)\}_{j=-1}^{N+1}$ spans S_Δ : Given $\Phi \in S_\Delta$ we need to show that

$$\Phi(x) = \sum_{j=-1}^{N+1} c_j \varphi_j(x).$$

Consider the interval $[x_j, x_{j+1}]$. The set $P_3(x_j, x_{j+1})$ of polynomials of degree 3 on $[x_j, x_{j+1}]$ is 4-dimensional. Since $\varphi_{j-1}, \varphi_j, \varphi_{j+1}, \varphi_{j+2} \in P_3(x_j, x_{j+1})$ are linearly independent, they span $P_3(x_j, x_{j+1})$ and thus form a basis for $P_3(x_j, x_{j+1})$. Now let $\Phi \in S_\Delta$, so Φ restricted to $[x_j, x_{j+1}]$ is in $P_3(x_j, x_{j+1})$. Thus,

$$\begin{aligned} \Phi(x) &= c_{j-1}^{(j)} \varphi_{j-1}(x) + c_j^{(j)} \varphi_j(x) + c_{j+1}^{(j)} \varphi_{j+1}(x) + c_{j+2}^{(j)} \varphi_{j+2}(x) \\ &\text{for } x \in [x_j, x_{j+1}], \end{aligned} \quad (4.31)$$

where the coefficients $c_{j+k}^{(j)}$, $k = -1, 0, 1, 2$ may depend on the interval j . If the coefficients can be shown to be independent of j , then every $\Phi \in S_\Delta$ can be written as a linear combination of $\varphi_j(x)$, $-1 \leq j \leq N+1$, for $x \in [a, b]$. To show this, consider $\Phi(x)$ on $[x_{j+1}, x_{j+2}]$:

$$\begin{aligned} \Phi(x) &= c_j^{(j+1)} \varphi_j(x) + c_{j+1}^{(j+1)} \varphi_{j+1}(x) + c_{j+2}^{(j+1)} \varphi_{j+2}(x) \\ &\quad + c_{j+3}^{(j+1)} \varphi_{j+3}(x) \quad \text{for } x \in [x_{j+1}, x_{j+2}]. \end{aligned} \quad (4.32)$$

If we show that $c_j^{(j)} = c_j^{(j+1)}$, $c_{j+1}^{(j)} = c_{j+1}^{(j+1)}$, $c_{j+2}^{(j)} = c_{j+2}^{(j+1)}$, then the c_j 's are independent of the interval. Equations (4.31) and (4.32) and C^2 -continuity of Φ at $x = x_{j+1}$ give

$$\begin{aligned} \Phi(x_{j+1}^-) &= \Phi(x_{j+1}^+) : \\ \frac{1}{6}c_j^{(j)} + \frac{2}{3}c_{j+1}^{(j)} + \frac{1}{6}c_{j+2}^{(j)} &= \frac{1}{6}c_j^{(j+1)} + \frac{2}{3}c_{j+1}^{(j+1)} + \frac{1}{6}c_{j+2}^{(j+1)} \\ \Phi'(x_{j+1}^-) &= \Phi'(x_{j+1}^+) : \\ -\frac{1}{2h}c_j^{(j)} + \frac{1}{2h}c_{j+2}^{(j)} &= -\frac{1}{2h}c_j^{(j+1)} + \frac{1}{2h}c_{j+2}^{(j+1)} \\ \Phi''(x_{j+1}^-) &= \Phi''(x_{j+1}^+) : \\ \frac{1}{h^2}c_j^{(j)} - \frac{2}{h^2}c_{j+1}^{(j)} + \frac{1}{h^2}c_{j+2}^{(j)} &= \frac{1}{h^2}c_j^{(j+1)} - \frac{2}{h^2}c_{j+1}^{(j+1)} + \frac{1}{h^2}c_{j+2}^{(j+1)} \end{aligned}$$

Thus,

$$\begin{pmatrix} 1 & 4 & 1 \\ -1 & 0 & 1 \\ 1 & -2 & 1 \end{pmatrix} \begin{pmatrix} c_j^{(j)} - c_j^{(j+1)} \\ c_{j+1}^{(j)} - c_{j+1}^{(j+1)} \\ c_{j+2}^{(j)} - c_{j+2}^{(j+1)} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

gives $c_j^{(j)} = c_j^{(j+1)}$, $c_{j+1}^{(j)} = c_{j+1}^{(j+1)}$ and $c_{j+2}^{(j)} = c_{j+2}^{(j+1)}$. $\square$

4.4.3 Cubic Spline Interpolants

Two types of cubic spline interpolant are commonly used: clamped boundary cubic spline interpolants and natural cubic spline interpolants.

DEFINITION 4.16 *The clamped boundary spline interpolant $\Phi_c \in S_\Delta$ of a function $f \in C^1[a, b]$ satisfies*

$$(c) \begin{cases} \Phi_c(x_i) = f(x_i), & i = 0, 1, \dots, N, \\ \Phi'_c(x_0) = f'(x_0), \\ \Phi'_c(x_N) = f'(x_N). \end{cases}$$

DEFINITION 4.17 *The natural spline interpolant $\Phi_n \in S_\Delta$ of a function $f \in C[a, b]$ satisfies*

$$(n) \begin{cases} \Phi_n(x_i) = f(x_i), & i = 0, 1, \dots, N, \\ \Phi''_n(x_0) = 0, \\ \Phi''_n(x_N) = 0. \end{cases}$$

PROPOSITION 4.7

Let $f \in C^1[a, b]$. Then there is a unique clamped boundary interpolant and a unique natural spline interpolant of f . (We are assuming here a uniform mesh.)

PROOF The proof is constructive, i.e., it describes how the interpolants Φ_c and Φ_n can be obtained.

Let

$$\Phi_c(x) = \sum_{j=-1}^{N+1} q_j \varphi_j(x).$$

The requirements (c) then lead to the system

$$\begin{cases} \sum_{j=-1}^{N+1} \varphi_j(x_i) q_j & = f(x_i), & i = 0, 1, \dots, N, \\ q_{-1} \varphi'_{-1}(x_0) + q_1 \varphi'_1(x_0) & = f'(x_0), \\ q_{N-1} \varphi'_{N-1}(x_N) + q_{N+1} \varphi'_{N+1}(x_N) & = f'(x_N), \end{cases}$$

since $\varphi'_0(x_0) = \varphi'_N(x_N) = 0$. The above system can be written in matrix form

as

$$\begin{pmatrix} 4 & 2 & & & \\ 1 & 4 & 1 & 0 & \\ & & \ddots & & \\ 0 & & & 1 & 4 & 1 \\ & & & 2 & 4 & \end{pmatrix} \begin{pmatrix} q_0 \\ q_1 \\ \vdots \\ q_{N-1} \\ q_N \end{pmatrix} = \begin{pmatrix} 6f(x_0) + 2hf'(x_0) \\ 6f(x_1) \\ \vdots \\ 6f(x_{N-1}) \\ 6f(x_N) - 2hf'(x_N) \end{pmatrix}, \quad (4.33)$$

where

$$q_{-1} = q_1 - 2hf'(x_0), \quad \text{and} \\ q_{N+1} = q_{N-1} + 2hf'(x_N).$$

The system (4.33) has a unique solution $\{q_j\}_{j=-1}^{N+1}$ because the matrix A is strictly diagonally dominant.

Now consider

$$\Phi_n(x) = \sum_{j=-1}^{N+1} s_j \varphi_j(x).$$

Conditions (n) lead to (exercise) the system

$$\begin{pmatrix} 6 & 0 & \cdots & & 0 \\ 1 & 4 & 1 & & \vdots \\ \vdots & & \ddots & & \\ & & & 1 & 4 & 1 \\ 0 & \cdots & & 0 & 6 & \end{pmatrix} \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{N-1} \\ s_N \end{pmatrix} = 6 \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_{N-1}) \\ f(x_N) \end{pmatrix}, \quad (4.34)$$

where

$$s_{-1} = -s_1 + 2s_0 \\ s_{N+1} = -s_{N-1} + 2s_N$$

Clearly, the s_j 's are uniquely determined. □

We give the following error estimate without proof.

THEOREM 4.15

Let $f \in C^4[a, b]$ and let $\Phi_c(x)$ be the clamped boundary cubic interpolant (uniform mesh). Then

$$\|f - \Phi_c\|_\infty \leq \frac{5}{384} h^4 \|D^4 f\|_\infty.$$

Furthermore,

$$\|D^r(f - \Phi_c)\|_\infty \leq c_r h^{4-r} \|D^4 f\|_\infty, \quad 0 \leq r \leq 3,$$

where $c_0 = 5/384$, $c_1 = 1/24$, $c_2 = 3/8$, and $c_3 = 1$.

PROOF See [80]. □

REMARK 4.37 A similar result holds for natural boundary cubic interpolants. See [14]. □

REMARK 4.38 Similar results hold for a nonuniform mesh. □

Example 4.12

Consider $f(x) = \ln x$. We wish to determine how small h should be to ensure that the cubic spline interpolant $\Phi_c(x)$ of $f(x)$ on the interval $[2, 4]$ has error less than 10^{-4} . We have

$$\|f - \Phi_c\|_\infty \leq \frac{5}{384} h^4 \|D^4 f\|_\infty = \frac{5}{384} h^4 \max_{2 \leq x \leq 4} \left| \frac{6}{x^4} \right| = \left(\frac{5}{384} \right) \left(\frac{6}{16} \right) h^4 \leq 10^{-4}.$$

Thus, $h^4 \leq (1/30)(384)(16)10^{-4}$, $h \leq 0.38$, and $N \geq 2/0.38 = 6$. (Recall that we required $N \geq 36$ to achieve the same error with piecewise linear interpolants.) □

REMARK 4.39 Satisfaction of $\Phi'_c(x_0) = f'(x_0)$, $\Phi'_c(x_N) = f'(x_N)$ may be difficult to achieve if $f(x)$ is not explicitly known. Approximations of order h^4 can then be used. Examples of such approximations are:

$$\begin{aligned} f'(x_0) &= \frac{1}{12h} \left[-25f(x_0) + 48f(x_0 + h) \right. \\ &\quad \left. - 36f(x_0 + 2h) + 16f(x_0 + 3h) - 3f(x_0 + 4h) \right] \\ &\quad + \langle \mathbf{error} \rangle, \\ &\quad \text{where } \langle \mathbf{error} \rangle = \frac{h^4}{5} f^{(5)}(\xi), \quad x_0 \leq \xi \leq x_0 + 4h, \\ f'(x_N) &= \frac{1}{12h} \left[25f(x_N) - 48f(x_N - h) + 36f(x_N - 2h) \right. \\ &\quad \left. - 16f(x_N - 3h) + 3f(x_N - 4h) \right] \\ &\quad + \langle \mathbf{error} \rangle, \\ &\quad \text{where } \langle \mathbf{error} \rangle = \frac{h^4}{5} f^{(5)}(\hat{\xi}), \quad x_N \leq \hat{\xi} \leq x_N - 4h. \end{aligned}$$

□

REMARK 4.40 It can be shown that if u is any C^2 -function on $[a, b]$

such that u interpolates f in the manner

$$\begin{cases} u(x_i) = f(x_i), & 0 \leq i \leq N, \\ u'(x_0) = f'(x_0), \\ u'(x_N) = f'(x_N), \end{cases}$$

then

$$\int_a^b (\Phi_c''(x))^2 dx \leq \int_a^b (u''(x))^2 dx.$$

That is, among all clamped C^2 -interpolants of f , the clamped spline interpolant is the smoothest in the sense of minimizing $\int_a^b (u''(x))^2 dx$. $\square$

We now turn to a type of approximation that has played a large role in classical applied mathematics, and is also important in modern signal processing technology, such as in storage and transmission of audio and video signals.

4.5 Trigonometric Approximation

Several good references on least squares trigonometric approximation and trigonometric interpolation are [31], [35], and [49].

4.5.1 Least Squares Trigonometric Approximation (Fourier Series)

Let $V = C[0, 2\pi]$ and

$$W_{2k} = \text{span}(w_{-k}, w_{-k+1}, \dots, w_0, w_1, \dots, w_k),$$

where $w_j(x) = e^{ijx}$ and $i = \sqrt{-1}$. Let

$$(f, g) = \int_0^{2\pi} f(x) \overline{g}(x) dx, \quad \text{for } f, g \in C[0, 2\pi].$$

Then, W_{2k} is a subspace of the inner product space V . It is straightforward to show that $(w_j, w_\ell) = 2\pi\delta_{j\ell}$. Thus, $\{w_{-k}, w_{-k+1}, \dots, w_0, w_1, \dots, w_k\}$ is an orthogonal basis for W_{2k} , and the best (least squares) approximation $g_k \in W_{2k}$ to $f \in V$ has the form

$$g_k(x) = \sum_{j=-k}^k \alpha_j e^{ijx}, \quad \text{where } \alpha_j = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ijx} dx. \quad (4.35)$$

DEFINITION 4.18

$$g_{\infty}(x) = \sum_{j=-\infty}^{\infty} \alpha_j e^{ijx}, \quad \text{where} \quad \alpha_j = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ijx} dx$$

is the complex Fourier series of $f(x)$, defined for $0 \leq x \leq 2\pi$.

REMARK 4.41 The complex Fourier series can be readily transformed to the standard form of Fourier series. Consider

$$\begin{aligned} g_k(x) &= \sum_{j=-k}^k \alpha_j e^{ijx} = \alpha_0 + \sum_{j=1}^k (\alpha_j e^{ijx} + \alpha_{-j} e^{ijx}) \\ &= \alpha_0 + \sum_{j=1}^k ((\alpha_j + \alpha_{-j}) \cos jx + i(\alpha_j - \alpha_{-j}) \sin jx) \\ &= a_0 + \sum_{j=1}^k (a_j \cos jx + b_j \sin jx), \end{aligned}$$

where

$$\begin{aligned} a_0 &= \alpha_0, \\ a_j &= \alpha_j + \alpha_{-j}, \quad \text{and} \\ b_j &= (\alpha_j - \alpha_{-j})i \quad \text{for } j = 1, 2, \dots, k. \end{aligned}$$

In addition, if a_0 , $\{a_j\}_{j=1}^k$, and $\{b_j\}_{j=1}^k$ are given, then

$$\begin{aligned} \alpha_0 &= a_0, \\ \alpha_j &= \frac{1}{2}(a_j - ib_j), \quad \text{and} \\ \alpha_{-j} &= \frac{1}{2}(a_j + ib_j) \quad \text{for } j = 1, 2, \dots, k. \end{aligned}$$

□

We begin our study of trigonometric approximation with a well-known result for Fourier series. To state this result, we first give

DEFINITION 4.19 Let $a = x_0 < x_1 < \dots < x_n = b$ be a subdivision of $[a, b]$. Define

$$t = \sum_{i=1}^n |f(x_i) - f(x_{i-1})|.$$

If $\sup t < \infty$, where the supremum is taken over all subdivisions of $[a, b]$, then f is said to be of bounded total variation. Notice, for example, if $f \in C^1[a, b]$, then f is of bounded total variation.

THEOREM 4.16

If $f(x)$ is piecewise continuous on $[0, 2\pi]$ and has bounded total variation, then

$$g_{\infty}(x) = \frac{1}{2}[f(x^+) + f(x^-)]$$

for $0 \leq x \leq 2\pi$, and $g_{\infty}(x)$ is repeated periodically outside the interval $[0, 2\pi]$. In particular,

$$g_{\infty}(0) = g_{\infty}(2\pi) = \frac{1}{2}[f(0^+) + f(2\pi^-)].$$

PROOF See [35]. □

REMARK 4.42 For the interval $[0, \pi]$ (rather than $[0, 2\pi]$ as above), the Fourier sine series of a function $f(x)$ defined for $0 < x < \pi$ is the function

$$g_S(x) = \sum_{\ell=1}^{\infty} a_{\ell} \sin \ell x, \quad \text{where} \quad a_{\ell} = \frac{2}{\pi} \int_0^{\pi} f(x) \sin \ell x dx. \quad (4.36)$$

The Fourier cosine series of a function defined for $0 < x < \pi$ is

$$g_C(x) = \sum_{\ell=0}^{\infty} a_{\ell} \cos \ell x, \quad (4.37)$$

where

$$a_0 = \frac{1}{\pi} \int_0^{\pi} f(x) dx \quad \text{and} \quad a_{\ell} = \frac{2}{\pi} \int_0^{\pi} f(x) \cos \ell x dx \quad \text{for } \ell \geq 1.$$

By using the change of variables $\tilde{x} = x - \pi$ in (4.35) and extending f evenly or oddly to $[-\pi, 0]$, it follows from Theorem 4.16 that if $f(x)$ is piecewise continuous and of bounded variation, then

$$g_S(x) = g_C(x) = \frac{1}{2}[f(x^+) + f(x^-)]$$

for $0 < x < \pi$. Furthermore, $g_S(-x) = -g_S(x)$ and $g_C(-x) = g_C(x)$ for $-\pi < x < 0$, and $g_S(x)$ and $g_C(x)$ are extended periodically outside $x \in [-\pi, \pi]$. □

We now have the following interesting result concerning the convergence of the least squares trigonometric approximation $g_k(x)$ to $f(x)$.

THEOREM 4.17

Suppose that $f \in C^n[0, 2\pi]$, $n \geq 2$, and

$$f^{(p)}(0) = f^{(p)}(2\pi)$$

for $p = 0, 1, \dots, n-1$. Then

$$\|g_k - f\|_\infty = \mathcal{O}(1/k^{n-1}).$$

In particular, if $f(x)$ is infinitely differentiable and periodic, then $g_k(x)$ converges to $f(x)$ more rapidly than any finite power of $1/k$ as $k \rightarrow \infty$.

PROOF By (4.36), then using integration by parts, we obtain

$$\begin{aligned} \alpha_j &= \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ijx} dx \\ &= - \left[\frac{1}{2\pi ij} f(x) e^{-ijx} \right]_0^{2\pi} + \frac{1}{2\pi ij} \int_0^{2\pi} f'(x) e^{ijx} dx \\ &\quad \vdots \end{aligned}$$

Thus, $\alpha_j = \frac{1}{2\pi (ij)^n} \int_0^{2\pi} f^{(n)}(x) e^{-ijx} dx$ by repeated integration by parts. Therefore,

$$\begin{aligned} |\alpha_j| &\leq \frac{1}{2\pi |j|^n} \int_0^{2\pi} |f^{(n)}(x)| |e^{-ijx}| dx \\ &\leq \max_{0 \leq x \leq 2\pi} |f^{(n)}(x)| \frac{1}{|j|^n} = \frac{c_n}{|j|^n} \text{ for } j = 0, \pm 1, \pm 2, \dots \end{aligned}$$

where $c_n = \max_{0 \leq x \leq 2\pi} |f^{(n)}(x)|$. By Theorem 4.16, $f(x) = g_\infty(x) = \sum_{j=-\infty}^{\infty} \alpha_j e^{ijx}$.

Thus,

$$\begin{aligned} \|f - g_k\|_\infty &= \max_{0 \leq x \leq 2\pi} \left| \sum_{j=-k}^k \alpha_j e^{ijx} - \sum_{j=-\infty}^{\infty} \alpha_j e^{ijx} \right| \\ &= \max_{0 \leq x \leq 2\pi} \left| \sum_{|j| > k} \alpha_j e^{ijx} \right| \leq \sum_{|j| > k} |\alpha_j| \\ &\leq 2c_n \sum_{j=k+1}^{\infty} \frac{1}{j^n}. \end{aligned}$$

Consider

$$\begin{aligned}
 \sum_{j=k+1}^{\infty} \frac{1}{j^n} &= \sum_{j=1}^{\infty} \frac{1}{(j+k)^n} = \frac{1}{k^n} \sum_{j=1}^{\infty} \frac{1}{(j/k+1)^n} \\
 &= \frac{1}{k^n} \left[\sum_{j=1}^k \frac{1}{(j/k+1)^n} + \sum_{j=k+1}^{2k} \frac{1}{(j/k+1)^n} \right. \\
 &\quad \left. + \sum_{j=2k+1}^{3k} \frac{1}{(j/k+1)^n} + \cdots \right] \\
 &\leq \frac{1}{k^n} \left[k + \frac{k}{2^n} + \frac{k}{3^n} + \cdots \right] = \frac{1}{k^{n-1}} \left[1 + \frac{1}{2^n} + \frac{1}{3^n} + \cdots \right] \\
 &= \frac{d_n}{k^{n-1}}, \text{ where } d_n = \sum_{j=1}^{\infty} \frac{1}{j^n} \text{ and } d_n \text{ is finite for } n \geq 2.
 \end{aligned}$$

Hence, $\|g_k - f\|_{\infty} \leq \frac{2c_n d_n}{k^{n-1}} = \mathcal{O}\left(\frac{1}{k^{n-1}}\right)$. □

Example 4.13

The L^2 -errors, i.e., $\|g_k - f\|_2$ for trigonometric approximations to various functions on $-1 \leq x \leq 1$ appear in the following table, where $f_1 \in C[-1, 1]$, $f_2 \in C^2[-1, 1]$, $f_3 \in C^{\infty}[-1, 1]$, $f_4 \in C^{\infty}[-1, 1]$, and f_4 is periodic.

k	$f_1(x) = x \log x $	$f_2(x) = x^2 e^{ x }$	$f_3(x) = \frac{1}{1+x^2}$	$f_4(x) = e^{\cos \pi x}$
2	0.053	0.22	0.014	0.045
4	0.030	0.097	0.0060	0.00054
8	0.015	0.038	0.0023	0.74×10^{-7}
16	0.0062	0.013	0.00083	—

For comparison, the calculated L^2 -errors for piecewise linear (PL) interpolants and cubic spline (CS) interpolants with M basis elements are given in the following table.

M	$f_1(x) = x \log x $		$f_2(x) = x^2 e^{ x }$		$f_3(x) = \frac{1}{1+x^2}$		$f_4(x) = e^{\cos \pi x}$	
	PL	CS	PL	CS	PL	CS	PL	CS
2	.38	.280	2.7	.065	.56	.011	3.0	.48
4	.46	.150	0.72	.0052	.16	.0021	1.4	.042
8	.25	.071	0.23	.00030	.039	.00016	0.41	.0085
16	.12	.030	0.068	.000019	.0088	.000071	0.10	.00031

□

4.5.2 Trigonometric Interpolation on a Finite Point Set (the FFT)

Trigonometric interpolation is often useful for interpolation of large amounts of data when the data are given at equally spaced points. In the trigonometric interpolation problem, we have N values of $f(x)$ on the interval $[0, 2\pi]$ at equally spaced points $x_j = 2\pi j/N$, $j = 0, 1, \dots, N-1$, i.e., $(x_0, f(x_0))$, $(x_1, f(x_1))$, $\dots$, $(x_{N-1}, f(x_{N-1}))$, where $f(0) = f(2\pi)$, i.e., f is periodic with period 2π . We wish to find the trigonometric polynomial

$$p(x) = \sum_{j=0}^{N-1} c_j e^{ijx}$$

that passes through these points. (See Figure 4.10.)

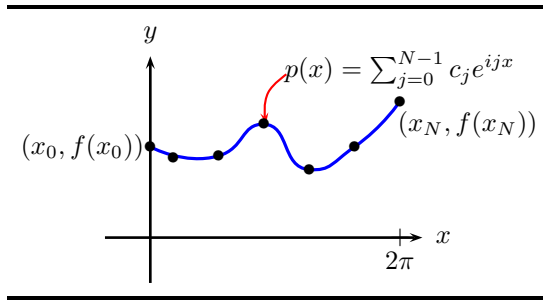


FIGURE 4.10: The graph of a trigonometric polynomial $p(x)$.

First, consider $\tilde{g}(x) = \sum_{j=0}^{N-1} \alpha_j e^{ijx}$, the best approximation to $f(x) \in C[0, 2\pi]$ in the inner product space with $(v, u) = \int_0^{2\pi} u(x) \overline{v(x)} dx$. (We assume that $f(x)$ is known.) Then, by (4.35), we know that

$$\tilde{g}(x) = \sum_{j=0}^{N-1} \alpha_j e^{ijx}, \quad \text{with} \quad \alpha_j = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ijx} dx. \quad (4.38)$$

Let's approximate the integral in (4.38) using the composite trapezoidal rule, i.e.,

$$\int_a^b f(x) dx \approx \frac{b-a}{N} \left[\frac{f(a) + f(b)}{2} + \sum_{j=1}^{N-1} f(x_j) \right],$$

where $x_j = a + jh$, $h = b - a/N$. Then,

$$\begin{aligned}\alpha_j &\approx \frac{1}{2\pi} \sum_{k=1}^{N-1} f(x_k) e^{-ijx_k} \frac{2\pi}{N} + \frac{2\pi}{N} \left[\frac{1}{2} f(x_0) e^{-ijx_0} + \frac{1}{2} f(x_N) e^{-ijx_N} \right] \\ &\approx \frac{1}{N} \sum_{k=0}^{N-1} f(x_k) e^{-ijx_k},\end{aligned}\tag{4.39}$$

assuming that $f(0) = f(x_0) = f(x_N) = f(2\pi)$, i.e., f is periodic. We will see that

$$c_j = \frac{1}{N} \sum_{k=0}^{N-1} f(x_k) e^{-ijx_k},\tag{4.40}$$

where $p(x) = \sum_{j=0}^{N-1} c_j e^{ijx}$ is the interpolating trigonometric polynomial. Thus, the interpolating trigonometric polynomial is closely related to the best approximating trigonometric polynomial in the inner product space.

To show that the c_j satisfy (4.40), we need the following lemma.

LEMMA 4.4

$$\frac{1}{N} \sum_{k=0}^{N-1} e^{-ijx_k} e^{i\ell x_k} = \begin{cases} 1 & \text{if } \ell = j, \\ 0 & \text{if } \ell \neq j, \end{cases}$$

where $x_k = 2\pi k/N$ and $0 \leq \ell, j \leq N-1$.

PROOF

$$\frac{1}{N} \sum_{k=0}^{N-1} e^{-ijx_k} e^{i\ell x_k} = \frac{1}{N} \sum_{k=0}^{N-1} e^{-i(\ell-j)\frac{2\pi k}{N}} = \frac{1}{N} \sum_{k=0}^{N-1} \lambda^k,$$

where $\lambda = e^{-i(\ell-j)2\pi/N}$. If $\ell = j$, then $\lambda = 1$ and thus

$$\frac{1}{N} \sum_{k=0}^{N-1} \lambda^k = 1.$$

If, on the other hand, $\ell \neq j$, then $\lambda \neq 1$, so

$$\frac{1}{N} \sum_{k=0}^N \lambda^k = \frac{\lambda^N - 1}{\lambda - 1} = \frac{e^{-2\pi i(\ell-j)} - 1}{e^{-i(\ell-j)\frac{2\pi}{N}} - 1} = 0,$$

where the last equation follows from the fact that $e^{-2\pi im} = 1$ for any integer m . $\square$

We now have:

THEOREM 4.18

The exponential polynomial

$$p(x) = \sum_{j=0}^{N-1} c_j e^{ijx} \quad \text{with} \quad c_j = \frac{1}{N} \sum_{k=0}^{N-1} f(x_k) e^{-ijx_k}$$

interpolates the points $\{(x_i, f(x_i))\}_{i=0}^{N-1}$.

PROOF

$$\begin{aligned} p(x_\nu) &= \sum_{j=0}^{N-1} c_j e^{ijx_\nu} = \frac{1}{N} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} f(x_k) e^{-ijx_k} e^{ijx_\nu} \\ &= \sum_{k=0}^{N-1} f(x_k) \left[\frac{1}{N} \sum_{j=0}^{N-1} e^{-ijx_k} e^{ijx_\nu} \right] = \sum_{k=0}^{N-1} f(x_k) \left[\frac{1}{N} \sum_{j=0}^{N-1} e^{ij(\nu-k)2\pi/N} \right] \\ &= \sum_{k=0}^{N-1} f(x_k) \left[\frac{1}{N} \sum_{j=0}^{N-1} e^{-ikx_j} e^{i\nu x_j} \right] = \sum_{k=0}^{N-1} f(x_k) \delta_{k,\nu} \\ &= f(x_\nu) \quad \text{for } \nu = 0, 1, 2, \dots, N-1. \end{aligned}$$

□

REMARK 4.43 Using Euler's identity, for real data $f(x_j) \in \mathbb{R}$ for $j = 0, 1, \dots, N-1$, $p(x) = \sum_{j=0}^{N-1} c_j e^{ijx}$ can be written in terms of a series of real functions $\cos jx$ and $\sin jx$ (Exercise 29 below). □

REMARK 4.44 To obtain

$$c_j = \frac{1}{N} \sum_{k=0}^{N-1} f(x_k) e^{-ijx_k}, \quad j = 0, 1, 2, \dots, N-1$$

directly requires $\mathcal{O}(N^2)$ operations, which is prohibitive for N large, say $N = 16384$. Fortunately, in 1965, Cooley and Tukey developed the Fast Fourier Transform (FFT) [19] for this purpose. This algorithm requires only $\mathcal{O}(N \log_2 N)$ operations. For example, for $N = 16384$, $N^2 = 2.684 \times 10^8$ and $N \log_2 N = 2.2938 \times 10^5$. The operation reduction in Fast Fourier Transform results from calculating the c_j in clusters. The FFT has enabled such modern technologies as compact disk audio, DVD video, storage and transmittal of sounds and video over the internet, and digital television signals. □

4.5.3 The FFT Procedure

Suppose $N = 2^r$ for some positive integer r . (This is not necessary, but the method is most efficient if $N = 2^r$, and some software requires this condition.) Let $M = N/2 = 2^{r-1}$. Then, for $j = 0, 1, 2, \dots, M-1$,

$$c_j = \frac{1}{2M} \sum_{k=0}^{2M-1} f(x_k) e^{-ijx_k} \quad \text{and}$$

$$c_{j+M} = \frac{1}{2M} \sum_{k=0}^{2M-1} f(x_k) e^{-ijx_k} e^{-iMx_k},$$

where $x_k = (2\pi k)/(2M) = (\pi k)/M$. Thus,

$$c_j + c_{j+M} = \frac{1}{2M} \sum_{k=0}^{2M-1} f(x_k) e^{-ij \frac{k\pi}{M}} (1 + e^{-i\pi k}),$$

but

$$(1 + e^{-i\pi k}) = \begin{cases} 2 & \text{if } k \text{ is even,} \\ 0 & \text{if } k \text{ is odd.} \end{cases}$$

Hence,

$$c_j + c_{j+M} = \frac{1}{M} \sum_{\substack{k=0 \\ k \text{ even}}}^{2M-1} f(x_k) e^{-ijk\pi/M} = \frac{1}{M} \sum_{k=0}^{M-1} f(x_{2k}) e^{-ij2k\pi/M} \quad (4.41)$$

for $j = 0, 1, \dots, M-1$. Similarly,

$$c_j - c_{j+M} = \frac{1}{M} \sum_{k=0}^{M-1} f(x_{2k+1}) e^{-ij(2k+1)\pi/M} \quad \text{for } j = 0, 1, 2, \dots, M-1. \quad (4.42)$$

The coefficients c_j and c_{j+M} for $j = 0, 1, \dots, M-1$ can be recovered from (4.41) and (4.42) with $\mathcal{O}(M^2)$ operations rather than the $\mathcal{O}((2M)^2)$ required in direct calculation of the c_j 's. Furthermore, letting, for $j = 0, 1, 2, \dots, M-1$,

$$\begin{cases} c_j^{(1,1)} = \frac{1}{M} \sum_{k=0}^{M-1} f(x_{2k}) e^{-ijk\pi/(M/2)} \\ c_j^{(1,2)} = \frac{1}{M} \sum_{k=0}^{M-1} f(x_{2k+1}) e^{-ij\pi/M} e^{-ijk\pi/(M/2)}, \end{cases} \quad (4.43)$$

we see that $c_j^{(1,1)}$ and $c_j^{(1,2)}$ are of the same form as

$$c_j = \frac{1}{2M} \sum_{k=0}^{2M-1} f(x_k) e^{-ijk\pi/M}.$$

Hence, the same procedure can be repeated on $c_j^{(1,1)}$ and $c_j^{(1,2)}$. If this process is repeated $(n + 1)$ times, it can be shown that the total number of operations is proportional to $N \log_2 N$. Schematically,

Level 1	Level 2	Level 4		Level $(r + 1)$
N constants	$N/2$ constants $N/2$ constants	$N/4$ constants $N/4$ constants $N/4$ constants $N/4$ constants	$\dots$	$\frac{N}{2^r}$ constants (2^r rows) $\frac{N}{2^r}$ constants
$\mathcal{O}(N^2)$ ops.	$2\mathcal{O}((\frac{N}{2})^2)$ ops.	$4\mathcal{O}((\frac{N}{4})^2)$ ops.		$2^r \mathcal{O}((\frac{N}{2^r})^2)$ ops.

REMARK 4.45 At each level, N constants are present. To go backward from Level $(i + 1)$ to level i , i.e., to find the constants at each level, requires $\mathcal{O}(N)$ operations per level. Thus, since r levels are present, the total number of operations is proportional to rN or $N \log_2 N$. $\square$

REMARK 4.46 The branch of applied analysis dealing with Fourier series is called *harmonic analysis*. The term is connected to the analysis of music and audio signals. For example, in a playing violin, a Fourier analysis of the function representing the air pressure produced by a vibrating string contains a primary frequency, then harmonics, representing twice the frequency, three times the frequency, etc. The function representing the vibration can be represented directly as a function $f(t)$, where t is time, or as the coefficients of its Fourier series $\{c_j\}_{j=-\infty}^{\infty}$. The time representation is called a representation in the *time domain*, while the representation in terms of the Fourier series is call the representation in the *frequency domain*. Replacing $\{c_j\}_{j=-\infty}^{\infty}$ by $\{c_j\}_{j=-N}^N$, or, more generally, dropping a selected set of coefficients c_j , is an example of *filtering*. $\square$

So far, we have studied approximation by polynomials, by piecewise polynomials, and by trigonometric functions. Polynomials and piecewise polynomials are appropriate for approximating bounded functions on closed intervals, while trigonometric functions are especially appropriate for approximating periodic functions. We next study rational functions, appropriate for approximating bounded functions on infinite intervals and functions with poles at points a (such as $f(x) = e^x/(x - a)$), as well as for more efficient approximations of continuous functions on closed intervals.

4.6 Rational Approximation

We now study approximation of functions by quotients of polynomials.

4.6.1 Introduction

DEFINITION 4.20 Let $R(L, M)$ denote the set of rational functions $r = r(x)$ of the form $r(x) = p(x)/q(x)$, where $p \in P_L$ and $q \in P_M$, i.e., p and q are polynomials of degree at most L and M , respectively.

We assume that $r(x)$ is irreducible, that is, that p and q have no common nonconstant factors. We consider in this section best rational approximants, Padé approximants, and interpolating rational functions. Good references for rational approximation include [17] and [75].

4.6.2 Best Rational Approximations in the Uniform Norm

THEOREM 4.19

If $f \in C[a, b]$, there exists an $r^* \in R(L, M)$ such that $\|f - r^*\|_\infty \leq \|f - r\|_\infty$ for all $r \in R(L, M)$, where $\|\cdot\|_\infty$ is the uniform norm on $C[a, b]$.

PROOF See [17]. (Note that $R(L, M)$ is not a finite-dimensional subspace of $C[a, b]$ so we cannot use the theory that we developed earlier.) $\square$

REMARK 4.47 An iterative procedure is used to compute r^* , since r^* cannot generally be found in a finite number of steps. Cheney has provided an early survey of such methods [16]. $\square$

4.6.3 Padé Approximation

Padé approximants are the rational function analogs of Taylor polynomial approximations. Padé approximants are useful in many areas, such as in numerical methods for solving ordinary differential equations. Rational functions are particularly useful when approximating functions with poles or when finding limiting values of functions as $x \rightarrow \pm\infty$.

Assume that $f \in C^{L+M+1}[-b, b]$, and suppose that $p(x)$ and $q(x)$ are polynomials of degree at most L and M , respectively, i.e., $p \in P_L$ and $q \in P_M$. Let

$$p(x) = \sum_{i=0}^L p_i x^i \quad \text{and} \quad q(x) = \sum_{j=0}^M q_j x^j.$$

DEFINITION 4.21 The L, M Padé approximant to $f(x)$ is

$$r(x) = \frac{p(x)}{q(x)}, \quad \text{such that}$$

$$f^{(k)}(0) = r^{(k)}(0) \quad \text{for } k = 0, 1, \dots, L + M, \text{ and}$$

$$q(0) = q_0 = 1.$$

(If $r(x)$ is irreducible and $q(0) = q_0 \neq 0$, we can divide the numerator and denominator by q_0 so that $q(0) = 1$.)

REMARK 4.48 If $M = 0$, then $r(x)$ is the Taylor polynomial approximation to $f(x)$ of degree at most L . $\square$

REMARK 4.49 The L, M Padé approximant as defined above may not exist.⁹ Consider, for example, $f(x) = 1 + x^2$, $M = 1$, and $L = 1$. Then $f(0) = 1$, $f'(0) = 0$, $f''(0) = 2$, and $r(x) = (p_0 + p_1x)/(1 + q_1x)$. $f(0) = 1$ gives $p_0 = 1$ and $f'(0) = 1$ gives $p_1 = q_1$. Hence, $r(x) = (1 + p_1x)/(1 + p_1x) = 1$ but $r''(0) = 0 \neq f''(0)$. Thus, in this example, the $1, 1$ Padé approximation does not exist. $\square$

PROPOSITION 4.8

Suppose that $f \in C^{L+M+1}[-b, b]$. The L, M Padé approximant $r(x)$ exists if and only if the coefficients p_i , $i = 0, 1, \dots, L$ and q_j , $j = 1, 2, \dots, M$ satisfy the equations

$$\sum_{\ell=0}^i \frac{f^{(\ell)}(0)}{\ell!} q_{i-\ell} = p_i \quad \text{for } i = 0, 1, 2, \dots, M + L, \quad (4.44)$$

where $p_i = 0$ for $i > L$, $q_j = 0$ for $j > M$, and $q_0 = 1$.

PROOF Suppose first that $r(x)$ exists. We can rule out denominators $q(x)$ that vanish at $x = 0$, since otherwise $r(x)$ is either reducible or is infinite at $x = 0$. We thus have $q_0 = 1$. Now consider

$$g(x) = (f(x) - r(x))q(x) = f(x)q(x) - p(x).$$

Since $f^{(k)}(0) - r^{(k)}(0) = 0$ for $k = 0, 1, 2, \dots, M + L$, it is straightforward to show that $g^{(k)}(0) = 0$ for $k = 0, 1, 2, \dots, M + L$. This implies that $g(x) = x^{M+L+1}Q(x)$, where $Q(x)$ is a continuous function of x , i.e., g has a zero of

⁹Other definitions for which the L, M Padé approximant does exist are sometimes used.

multiplicity $M + L + 1$ at $x = 0$. (For example, suppose that $g \in C^1[-b, b]$ and $g(0) = 0$. Let $g'(x) = v(x)$. Then

$$\int_0^x g'(s)ds = g(x) = \int_0^x v(s)ds,$$

so $g(x) = xQ(x)$, where $Q(x) = \frac{1}{x} \int_0^x v(s)ds$ is a continuous function of x .) Now,

$$g(x) = f(x)q(x) - p(x) = \sum_{\ell=0}^{M+L} a_{\ell} x^{\ell} \sum_{j=0}^M q_j x^j - \sum_{i=0}^L p_i x^i + R(x)q(x),$$

where

$$a_{\ell} = \frac{f^{(\ell)}(0)}{\ell!}$$

and

$$R(x) = \int_0^x \frac{(x-t)^{M+L}}{(M+L)!} f^{(M+L+1)}(t)dt = f^{(M+L+1)}(\xi) \frac{x^{M+L+1}}{(M+L+1)!},$$

for some ξ between 0 and x . Thus, for $g(x) = x^{M+L+1}Q(x)$ to be true, we must have

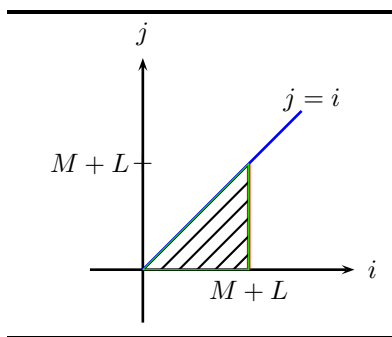
$$\sum_{\ell=0}^{M+L} \sum_{j=0}^M a_{\ell} q_j x^{\ell+j} - \sum_{i=0}^L p_i x^i = 0 \quad \text{for powers of } x \text{ up to } L+M. \quad (4.45)$$

But
$$\sum_{\ell=0}^{M+L} \sum_{j=0}^M a_{\ell} q_j x^{\ell+j}$$

$$= \sum_{j=0}^M \sum_{\ell=0}^{M+L} a_{\ell} q_j x^{\ell+j} = \sum_{j=0}^M \sum_{i=j}^{M+L+j} a_{i-j} q_j x^i \quad (\text{letting } i = \ell + j)$$

$$= \sum_{j=0}^{M+L} \sum_{i=j}^{M+L} a_{i-j} q_j x^i \quad \text{for powers of } x \text{ up to } L+M \text{ (since } q_j = 0 \text{ for } j > M)$$

$$= \sum_{i=0}^{M+L} \sum_{j=0}^i a_{i-j} q_j x^i \quad \text{switching sums (see the figure below).}$$



Thus, by (4.45),

$$p_i = \sum_{j=0}^i a_{i-j} q_j \quad \text{for } i = 0, 1, \dots, M + L$$

(by equating like powers of x). Thus, by the change of index $i - j = \ell$,

$$p_i = \sum_{\ell=0}^i a_{\ell} q_{i-\ell} \quad \text{or} \quad p_i = \sum_{\ell=0}^i \frac{f^{(\ell)}(0)}{\ell!} q_{i-\ell}$$

for $i = 0, 1, \dots, M + L$.

Conversely, given that (4.44) is satisfied, it is straightforward to show that $r^{(k)}(0) = f^{(k)}(0)$ for $k = 0, 1, \dots, M + L$. $\square$

Example 4.14

Find the 1,1 Padé approximant of $f(x) = e^x$.

Solution: $q_0 = 1$, $f^{(\ell)}(0) = 1$,

$$\sum_{\ell=0}^i \frac{1}{\ell!} q_{i-\ell} = p_i$$

for $i = 0, 1, 2$. Thus, for $i = 0$, $p_0 = 1$. For $i = 1$, $q_1 + q_0 = p_1$. For $i = 2$, $q_2 + q_1 + \frac{1}{2}q_0 = p_2$, but $q_2 = p_2 = 0$. Solving, $p_0 = 1$, $q_1 = -\frac{1}{2}$, and $p_1 = \frac{1}{2}$. Hence,

$$e^x \approx r(x) = \frac{1 + x/2}{1 - x/2}.$$

Table 4.3 compares this approximation to the second degree Taylor polynomial at several points. $\square$

We now have the following interesting result:

PROPOSITION 4.9

If $f(x)$ has $L + M + 1$ continuous derivatives in $[-b, b]$ and the L, M Padé approximant $p(x)/q(x)$ exists, then the $(L + M)$ -th degree Maclaurin polynomial

TABLE 4.3: Comparison of the Padé approximant of Example 4.14 to a degree-2 Taylor polynomial

x	$r(x)$	$1 + x + \frac{x^2}{2}$ (Taylor series)	e^x
0	1.000	1.000	1.0000
1/2	1.667	1.625	1.6487
-1/2	0.600	0.625	0.6065

of $f(x)q(x)s(x)$ is equal to $p(x)s(x)$, where $s(x)$ is an arbitrary polynomial of degree M .

PROOF Let $s(x) = \sum_{i=0}^M s_i x^i$ and let $v(x)$ be the $(L+M)$ -th degree Maclaurin polynomial of $f(x)q(x)s(x)$. Then

$$\begin{aligned}
 v(x) &= \sum_{j=0}^{L+M} \left[f(x)q(x)s(x) \right]_{x=0}^{(j)} \frac{x^j}{j!} \\
 &= \sum_{j=0}^{L+M} \sum_{i=0}^j \binom{j}{i} s^{(j-i)}(0) \left[(f(x)q(x)) \right]_{x=0}^{(i)} \frac{x^j}{j!} \\
 &= \sum_{j=0}^{L+M} \sum_{i=0}^j \binom{j}{i} s^{(j-i)}(0) \sum_{\ell=0}^i f^{(\ell)}(0) q^{(i-\ell)}(0) \binom{i}{\ell} \frac{x^j}{j!} \\
 &= \sum_{j=0}^{L+M} \sum_{i=0}^j \sum_{\ell=0}^i f^{(\ell)}(0) q^{(i-\ell)}(0) s^{(j-i)}(0) \frac{j!}{(j-i)!i!} \frac{i!}{(i-\ell)! \ell!} \frac{x^j}{j!} \\
 &= \sum_{j=0}^{L+M} \sum_{i=0}^j \sum_{\ell=0}^i \frac{f^{(\ell)}(0) q^{(i-\ell)}(0) s^{(j-i)}(0)}{\ell!(i-\ell)!(j-i)!} x^j \\
 &= \sum_{j=0}^{L+M} \sum_{i=0}^j \sum_{\ell=0}^i \frac{f^{(\ell)}(0)}{\ell!} q_{i-\ell} \frac{s^{(j-i)}(0)}{(j-i)!} x^j \\
 &= \sum_{j=0}^{L+M} \sum_{i=0}^j p_i s_{j-i} x^j \text{ (from (4.44))} \\
 &= p(x)s(x),
 \end{aligned}$$

since p and s are L -th and M -th degree polynomials, respectively. $\square$

REMARK 4.50 The *Leibniz rule* was used in the proof of Proposition 4.9.

The Leibnitz rule is

$$(fg)^{(n)} = \sum_{k=0}^n \binom{n}{k} f^{(k)} g^{(n-k)},$$

where $f^{(k)}$ denotes the k -th derivative of the function f . □

REMARK 4.51 By Proposition 4.9, $p(x)s(x)$ is the $(L + M)$ -th degree Taylor polynomial about zero of $f(x)q(x)s(x)$. By considering the remainder term in Taylor polynomial approximation, we can obtain an error estimate for Padé approximation. We have

$$f(x)q(x)s(x) - p(x)s(x) = \frac{x^{L+M+1}}{(L + M + 1)!} \left[\frac{d^{L+M+1}}{dx^{L+M+1}} (f(x)q(x)s(x)) \right]_{x=\xi}.$$

Thus,

$$f(x) - \frac{p(x)}{q(x)} = \frac{x^{L+M+1}}{s(x)q(x)(L + M + 1)!} \frac{d^{L+M+1}(fqs)(\xi)}{dx^{L+M+1}},$$

or

$$\left| f(x) - \frac{p(x)}{q(x)} \right| \leq \frac{x^{L+M+1}}{|s(x)q(x)|(L + M + 1)!} M^* \quad (4.46)$$

where

$$M^* = \max_{-b \leq x \leq b} \left| \frac{d^{L+M+1}}{dx^{L+M+1}} (f(x)q(x)s(x)) \right|,$$

and where s is any polynomial of degree at most M . For example, s can be the constant $s \equiv 1$. □

Example 4.15

Consider $f(x) = \ln(1 + x)$ for $x \geq 0$. The $L = 2$, $M = 1$ Padé approximant to $f(x)$ is

$$r(x) = \frac{6x + x^2}{6 + 4x}.$$

Also,

$$\max_{x \geq 0} \left| \frac{d^4}{dx^4} (fq) \right| = \max_{x \geq 0} \left| \frac{8}{(1+x)^3} - \frac{12}{(1+x)^4} \right| = 4.$$

Consider $s(x) = 1$. Then

$$\left| f(x) - \frac{p(x)}{q(x)} \right| \leq \frac{x^4 4}{(6 + 4x)4!} = \frac{x^4}{36 + 24x}$$

for $x \geq 0$. For $x = 0.1$,

$$\left| f(x) - \frac{p(x)}{q(x)} \right| \leq 2.6 \times 10^{-6}.$$

In contrast, for the Taylor polynomial approximation of degree 3 at $x = 0.1$,

$$\left| f(x) - x + \frac{x^2}{2} - \frac{x^3}{3} \right| = 2.5 \times 10^{-5},$$

and the Taylor polynomial of degree 3 requires same computational work to evaluate but is a factor of 10 less accurate than the 2,1 Padé approximant at $x = 0.1$. $\square$

Padé approximations are commonly used for $f(x) = e^x$. (We will see their use in the study of numerical methods for solving initial-value problems.) Table 4.4 gives a few of these Padé approximants.

TABLE 4.4: Padé approximants to e^x

	$M = 0$	$M = 1$	$M = 2$
$L = 0$	1	$\frac{1}{1-x}$	$\frac{1}{1-x+x^2/2}$
$L = 1$	$1+x$	$\frac{1+x/2}{1-x/2}$	$\frac{1+x/3}{1-(2/3)x+x^2/6}$
$L = 2$	$1+x+x^2/2$	$\frac{1+(2/3)x+x^2/6}{1-x/3}$	$\frac{1+x/2+x^2/12}{1-x/2+x^2/12}$

REMARK 4.52 To find Padé approximants to e^{-x} , replace x by $-x$ in Table 4.4. $\square$

REMARK 4.53 A precise error formula for $f(x) = e^x$ is

$$\begin{aligned} e^x - \frac{p(x)}{q(x)} &= \frac{1}{(M+L)!q(x)} \int_0^x (x-t)^L (-t)^M e^t dt \\ &= \frac{e^x (-1)^M x^{L+M+1}}{(M+L)!q(x)} \int_0^1 z^L (1-z)^M e^{-xz} dz \end{aligned}$$

(letting $z = 1 - t/x$ and $t = x - xz$). $\square$

4.6.4 Rational Interpolation

Recall that $r \in R(L, M)$ if $r(x) = p(x)/q(x)$, where $p \in P_L$ and $q \in P_M$.

Problem: Find $r \in R(L, M)$ such that $r(x_i) = y_i$, $1 \leq i \leq k$, where x_i , $1 \leq i \leq k$, are distinct points on $[a, b]$.

Difficulty: $r(x)$ may not exist, as the following two examples illustrate.

Example 4.16

Suppose that $r \in R(0, M)$. Then, $r(x) = 1/q(x)$. Suppose that $y_i = 0$ for some i . Then we cannot interpolate this point. $\square$

Example 4.17

Suppose that $r \in R(1, 1)$, i.e.,

$$r(x) = \frac{a_0 + x}{b_0 + b_1 x}.$$

Suppose that $y_1 = y_2 \neq y_3$. Then $r(x_1) = r(x_2) = y_1 = y_2$, and it follows that

$$\frac{x_1 + a_0}{b_0 + b_1 x_1} = \frac{x_2 + a_0}{b_0 + b_1 x_2},$$

and thus, $b_0(x_1 - x_2) = a_0 b_1(x_1 - x_2)$, or $b_0 = a_0 b_1$.

Case a: If $b_1 \neq 0$, then $r(x) = \frac{a_0 + x}{b_1(a_0 + x)} = \frac{1}{b_1}$. Thus, $r(x)$ is constant, and $r(x_3) \neq y_3$.

Case b: If $b_1 = 0$, then $b_0 = 0$ and the denominator vanishes, so $r(x)$ does not exist. $\square$

Example 4.18

Suppose that $r \in R(L, 0)$. Then the interpolation problem has a unique solution when $K = L + 1$. Indeed, $r(x)$ is the Lagrange interpolating polynomial. $\square$

If $r(x)$ does exist, we have the following.

THEOREM 4.20

Suppose that f has k continuous derivatives on $[a, b]$ and suppose that $a = x_1 < x_2 < \cdots < x_k = b$. Let $r = p/q \in R(L, M)$, where $k = L + M + 1$, interpolates $f(x_i)$ at x_i , $1 \leq i \leq k$. Then for each $x \in [a, b]$, there exists a $\xi = \xi(x) \in [a, b]$ such that

$$f(x) - \frac{p(x)}{q(x)} = \frac{(x - x_1)(x - x_2) \cdots (x - x_k)}{k!q(x)} \frac{d^k(fq)}{dx^k}(\xi) \quad (4.47)$$

where we assume that $q(x)$ does not vanish on $[a, b]$. As a consequence,

$$\max_{a \leq x \leq b} \left| f(x) - \frac{p(x)}{q(x)} \right| \leq \max_{a \leq x \leq b} \left| \frac{\prod_{i=1}^k (x - x_i)}{k!q(x)} \right| \max_{a \leq x \leq b} \left| \frac{d^k f(x)q(x)}{dx^k} \right|$$

PROOF If

$$f(x_i) - \frac{p(x_i)}{q(x_i)} = 0 \quad \text{for } 1 \leq i \leq k,$$

then

$$f(x_i)q(x_i) - p(x_i) = 0 \quad \text{for } 1 \leq i \leq k, \quad (4.48)$$

since $q(x_i) \neq 0$ for $i = 1, 2, \dots, k$. Hence, $p(x)$ is the Lagrange interpolating polynomial of $f(x)q(x)$ for points x_i , $1 \leq i \leq k$. Thus we know for Lagrange interpolation,

$$f(x)q(x) - p(x) = \frac{(x - x_1)(x - x_2) \dots (x - x_k)}{k!} \frac{d^k(fq)(\xi)}{dx^k}$$

for some $\xi(x) \in [a, b]$. Dividing by $q(x)$, we obtain the desired result. $\square$

Next, we consider a type of orthogonal basis with certain mathematical properties similar to those of the Fourier series basis ($\sin(kx)$ and $\cos(kx)$, or e^{ikx}). Various bases in this class can be designed to do a good job at particular approximation tasks.

4.7 Wavelet Bases

In this section, we give a brief description of wavelet bases in $L^2(\mathbb{R})$. Two good references are [3] and [15]. Wavelets are special sets of orthogonal functions in $L^2(\mathbb{R})$, where $L^2(\mathbb{R})$ is the set of integrable functions f such that

$$\int_{-\infty}^{\infty} f^2(x)dx < \infty,$$

and where orthogonality is with respect to the inner product

$$(f, g) = \int_{-\infty}^{\infty} f(x)g(x)dx.$$

Wavelets are distinguished by being able to accurately represent high-frequency signals. (Because wavelets are orthogonal basis functions on the inner product space $L^2(\mathbb{R})$, the least squares approximation in terms of a wavelet expansion is easy to obtain.)

4.7.1 Assumed Properties of the Scaling Function

Underlying the development of wavelets is a “scaling function” $\varphi(x) \in L^2(\mathbb{R})$. We assume that φ satisfies:

$$\varphi(x) = \sum_{\ell=0}^L a_{\ell} \varphi(2x - \ell) \quad (4.49)$$

for some real numbers a_{ℓ} , where $\varphi(x) = 0$ for $x < 0$ or $x \geq L$, i.e., φ is supported on $[0, L)$. For notational convenience, we also assume that the expansion in (4.49) is infinite, but $a_{\ell} = 0$ for $\ell < 0$ or $\ell > L$. In addition, assume that

$$\int_{-\infty}^{\infty} \varphi(x - k) \varphi(x - \ell) dx = \delta_{k\ell}. \quad (4.50)$$

Example 4.19

$L = 1$, $a_0 = 1$, and $a_1 = 1$. Then $\varphi(x) = \varphi(2x) + \varphi(2x - 1)$, and $\varphi(x)$ is supported on $[0, 1)$. This implies that φ is constant on $[0, 1)$. Hence, by (4.50), $\varphi(x) = 1$ for $x \in [0, 1)$. Note that

$$\varphi(x - k) = \begin{cases} 1 & \text{if } k \leq x < k + 1 \\ 0 & \text{otherwise.} \end{cases}$$

□

PROPOSITION 4.10

Equations (4.49) and (4.50) give the condition

$$\frac{1}{2} \sum_{m=-\infty}^{\infty} a_{2k+m} a_{2\ell+m} = \delta_{k\ell}, \quad -\infty < k, \ell < \infty,$$

PROOF

$$\begin{aligned} \delta_{k\ell} &= \int_{-\infty}^{\infty} \varphi(x - k) \varphi(x - \ell) dx \\ &= \int_{-\infty}^{\infty} \sum_{m=-\infty}^{\infty} a_m \varphi(2(x - k) - m) \sum_{n=-\infty}^{\infty} a_n \varphi(2(x - \ell) - n) dx \\ &= \frac{1}{2} \sum_m \sum_n a_m a_n \int_{-\infty}^{\infty} \varphi(2x - 2k - m) \varphi(2x - 2\ell - n) 2dx \\ &= \frac{1}{2} \sum_m \sum_n a_m a_n \delta_{2k+m, 2\ell+n} = \frac{1}{2} \sum_m a_m a_{2(k-\ell)+m} \\ &= \frac{1}{2} \sum_{\hat{m}=-\infty}^{\infty} a_{\hat{m}+2\ell} a_{\hat{m}+2k} \quad (\text{by letting } \hat{m} = m - 2\ell.) \end{aligned}$$

□

Example 4.20

Let φ be as in Example 4.19. Note that $L = 1$, $a_0 = 1$, $a_1 = 1$ satisfies Proposition 4.10. Specifically,

$$\begin{aligned} \frac{1}{2} \sum_{m=-\infty}^{\infty} a_{2k+m} a_{2\ell+m} &= \frac{1}{2} a_0 a_{2\ell-2k} + \frac{1}{2} a_1 a_{2\ell-2k+1} \\ &= \frac{1}{2} a_{2\ell-2k} + \frac{1}{2} a_{2\ell-2k+1} \\ &= \delta_{k\ell}. \end{aligned}$$

□

Now let $V_0 \subset L^2(\mathbb{R})$ be the set of integer translates of φ , i.e.,

$$V_0 = \left\{ f \in L^2(\mathbb{R}) : f(x) = \sum_{k=-\infty}^{\infty} \alpha_k \varphi(x-k) \right\}. \quad (4.51)$$

Then V_0 is a closed subspace of $L^2(\mathbb{R})$. In addition,

$$\alpha_k = \int_{-\infty}^{\infty} f(x) \varphi(x-k) dx$$

by the orthogonality property (4.50).

Example 4.21

If φ is as in Example 4.20, then $f \in V_0$ has the form shown in Figure 4.11.

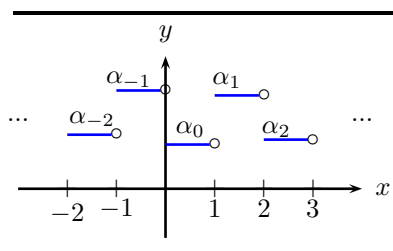


FIGURE 4.11: The graph of a piecewise constant function f , as in Example 4.21.

We now define V_n as *dilates* of V_0 , i.e.,

$$V_n = \left\{ f \in L^2(\mathbb{R}) : f(x) = \sum_{k=-\infty}^{\infty} \alpha_k \varphi(2^{-n}x - k) \right\}. \quad (4.52)$$

□

REMARK 4.54 Note that $\varphi(2^{-n}x)$ has support $[0, 2^n L)$ and $\varphi(2^{-n}x - k)$ has support $[2^n k, 2^n k + 2^n L)$. Also, if $f \in V_n$, then

$$\alpha_k = \frac{\int_{-\infty}^{\infty} f(x) \varphi(2^{-n}x - k) dx}{\int_{-\infty}^{\infty} \varphi^2(2^{-n}x - k) dx} = 2^{-n} \int_{-\infty}^{\infty} f(x) \varphi(2^{-n}x - k) dx.$$

□

REMARK 4.55 Notice that if $f \in V_n$ then $f \in V_{n-1}$. We thus have the containment hierarchy:

$$\cdots \subset V_3 \subset V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \subset \cdots \subset L^2(\mathbb{R}). \quad (4.53)$$

To see this more clearly, suppose for example that $f \in V_0$. Then

$$f(x) = \sum_{k=-\infty}^{\infty} \alpha_k \varphi(x - k) = \sum_{k=-\infty}^{\infty} \alpha_k \sum_{\ell=0}^L \varphi(2x - 2k - \ell).$$

Hence,

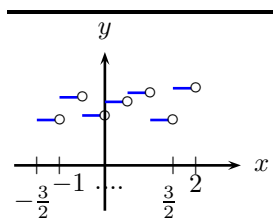
$$\begin{aligned} f(x) &= \sum_{\ell=0}^L a_{\ell} \sum_{k=-\infty}^{\infty} \alpha_k \varphi(2x - 2k - \ell) \\ &= \sum_{\ell=0}^L a_{\ell} \left(\sum_{\hat{k}=-\infty}^{\infty} \alpha_{\frac{\hat{k}-\ell}{2}} \varphi\left(2x - \hat{k}\right) \right) \quad (\text{substituting } \hat{k} = 2k + \ell) \\ &= \sum_{\ell=0}^L a_{\ell} g_{\ell}(x) \in V_{-1} \quad (\text{where } (\hat{k} - \ell)/2 \text{ is an integer.}) \end{aligned}$$

Thus, if $f \in V_0$ then $f \in V_{-1}$.

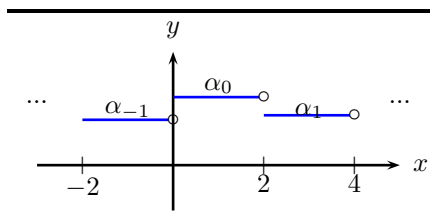
□

Example 4.22

For Example 4.21, $f \in V_{-1}$ has the form shown in the following figure,



while $f \in V_1$ has the form shown in the following figure.



□

We now make one final assumption concerning the scaling function $\varphi(x)$: Assume that

$$x^j \in V_0 \text{ for } j = 0, 1, \dots, N-1. \quad (4.54)$$

That is,

$$x^j = \sum_{k=-\infty}^{\infty} \alpha_{j,k} \varphi(x-k)$$

for some $\{\alpha_{j,k}\}_{k=-\infty}^{\infty}$.

REMARK 4.56 N and L are related to each other.

□

Example 4.23

For Example 4.22, with $L = 1$, $\alpha_0 = 1$, $\alpha_1 = 1$, and we see that $N = 1$. That is, $1 \in V_0$, but $x \notin V_0$. □

The following convergence result can now be shown for least-squares approximations in V_n .

THEOREM 4.21

Suppose that (4.49) and (4.50) are satisfied for positive integers L and N . Let $f \in C^N(\mathbb{R}) \cap L^2(\mathbb{R})$ and let $f_j \in V_j$ be the least-squares approximation to f in V_j , i.e.,

$$f_j(x) = \sum_{k=-\infty}^{\infty} \alpha_{j,k} \varphi(2^{-j}x - k) \quad \text{with} \quad \alpha_{j,k} = 2^{-j} (f, \varphi(2^{-j}x - k)).$$

Then the sequence, $f_1, f_0, f_{-1}, \dots$ converges to f with order N .

PROOF See [87]. □

REMARK 4.57 Theorem 4.21 implies that the union of the V_n is dense in $L^2(\mathbb{R})$, i.e., $\overline{\bigcup_n V_n} = L^2(\mathbb{R})$. □

Example 4.24

For Example 4.21, $L = 1$ and $N = 1$. Thus, the sequence $f_2, f_1, f_0, f_{-1}, \dots$ converges to $f \in C^1(\mathbb{R}) \cap L^2(\mathbb{R})$ with order 1. □

4.7.2 The Wavelet Basis and the Scaling Function

We now develop an orthogonal basis in $L^2(\mathbb{R})$. Let W_n be the orthogonal complement of V_n in V_{n-1} . That is,

$$W_n \cup V_n = V_{n-1} \text{ and } W_n \perp V_n. \quad (4.55)$$

Thus, if $f \in V_{n-1}$ then $f = f_1 + f_2$ where $f_1 \in W_n$, $f_2 \in V_n$, and $(f_1, f_2) = 0$.

REMARK 4.58 Suppose that $n < m$. Then $W_m \cup V_m = V_{m-1}$, $W_m \perp V_m$, $W_n \cup V_n = V_{n-1}$, and $W_n \perp V_n$. But $W_m \subset V_{m-1} \subset V_n$, so $W_m \perp W_n$. Thus, if $n \neq m$, $W_n \perp W_m$. This implies, along with Remark 4.57, that the direct sum of the W_n is $L^2(\mathbb{R})$. Hence,

$$\bigcup_{n=-\infty}^{\infty} W_n = \overline{\bigcup_{n=-\infty}^{\infty} V_n} = L^2(\mathbb{R}) \text{ and } W_n \perp W_m \text{ for } n \neq m. \quad (4.56)$$

□

REMARK 4.59 Recall that V_0 is spanned by integer translates of $\varphi(x)$. V_{-1} is spanned by integer translates of $\varphi(2x)$ and $\varphi(2x-1)$. Since $V_0 \cup W_0 = V_{-1}$, this implies that W_0 is spanned by integer translates of some function $w(x)$. Thus,

$$W_0 = \left\{ f \in L^2(\mathbb{R}) : f(x) = \sum_{k=-\infty}^{\infty} \alpha_k w(x-k) \right\}.$$

Similarly, the spaces W_n , $-\infty < n < \infty$, are dilates of W_0 . Thus, from (4.56),

$$L^2(\mathbb{R}) = \left\{ f : f(x) = \sum_{n=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \alpha_{n,k} w(2^{-n}x - k) \right\}.$$



DEFINITION 4.22 *The function $w(x)$ is called a wavelet. (Dilates and translates of $w(x)$ are orthogonal and form a basis for $L^2(\mathbb{R})$.) We now wish to determine $w(x)$ from the scaling function $\varphi(x)$. Note that $w \in W_0 \subset V_{-1}$ so*

$$w(x) = \sum_{k=-\infty}^{\infty} b_k \varphi(2x - k). \quad (4.57)$$

However, recall that $\varphi \in V_0 \subset V_{-1}$, so

$$\varphi(x) = \sum_{k=-\infty}^{\infty} a_k \varphi(2x - k) = \sum_{k=0}^L a_k \varphi(2x - k). \quad (4.58)$$

The following proposition shows that $b_k = (-1)^k a_{1-k}$, and thus

$$w(x) = \sum_{k=1}^{-L+1} (-1)^k a_{1-k} \varphi(2x - k).$$

PROPOSITION 4.11

If $b_k = (-1)^k a_{1-k}$, then

$$\int_{-\infty}^{\infty} \varphi(x - k) w(x - \ell) dx = 0 \quad \text{and} \quad \int_{-\infty}^{\infty} w(x - k) w(x - \ell) dx = \delta_{k\ell}.$$

(This guarantees that $W_0 \perp V_0$.)

PROOF

$$\begin{aligned}
\int_{-\infty}^{\infty} \varphi(x-k)w(x-\ell)dx &= \int_{-\infty}^{\infty} \frac{1}{2} \left[\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} a_m b_n \cdot \right. \\
&\quad \left. \varphi(2x-2k-m)\varphi(2x-2\ell-n) \right] 2dx \\
&= \frac{1}{2} \sum_{m=-\infty}^{\infty} a_{m+2\ell} b_{m+2k} \quad (\text{See the proof of Proposition 4.10.}) \\
&= \frac{1}{2} \sum_{m=-\infty}^{\infty} a_{m+2\ell} (-1)^{2k+m} a_{1-2k-m} \\
&= 0 \quad (\text{since each product } a_i a_j \text{ occurs twice} \\
&\quad \text{with opposite signs}) \\
\int_{-\infty}^{\infty} w(x-k)w(x-\ell)dx &= \int_{-\infty}^{\infty} \left[\sum_{m=-\infty}^{\infty} b_m \varphi(2x-2k-m) \cdot \right. \\
&\quad \left. \sum_{n=-\infty}^{\infty} b_n \varphi(2x-2\ell-n) \right] dx \\
&= \frac{1}{2} \sum_{m=-\infty}^{\infty} b_{2k+m} b_{2\ell+m} \\
&= \frac{1}{2} \sum_{m=-\infty}^{\infty} (-1)^{2k+m} a_{1-2k-m} (-1)^{2\ell+m} a_{1-2\ell-m} \\
&= \frac{1}{2} \sum_{m=-\infty}^{\infty} a_{1-2k-m} a_{1-2\ell-m} \\
&= \delta_{k\ell}, \quad \text{by Proposition (4.10).}
\end{aligned}$$

□

Example 4.25

Continuing Example 4.21, $L = 1$, $a_0 = 1$, $a_1 = 1$, so $b_1 = -1$ and $b_0 = 1$. Thus, $w(x) = \varphi(2x) - \varphi(2x-1)$; see Figure 4.12. That is,

$$w(x) = \begin{cases} 1, & 0 \leq x \leq 1/2 \\ -1, & 1/2 \leq x \leq 1 \\ 0, & \text{otherwise.} \end{cases}$$

The basis constructed using this $w(x)$ is called the *Haar basis*. □

The Haar basis is easily constructed and is the simplest wavelet basis. This basis has $L = 1$ and $N = 1$, and Theorem 4.21 shows that the approximations

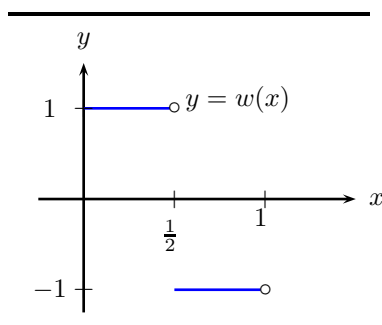


FIGURE 4.12: $w(x) = \varphi(2x) - \varphi(2x - 1)$ in Example 4.25.

converge in this basis, but not particularly fast. To improve the convergence rate, an $N \geq 2$ would be required. However, in this case the scaling function $\varphi(x)$ is complicated. To determine $\varphi(x)$, we need the following additional relation.

PROPOSITION 4.12

$\sum_{k=-\infty}^{\infty} (-1)^k a_k k^j = 0$ for $j = 0, 1, \dots, N - 1$, where the a_k are as in (4.49), subject to the additional assumption (4.54).

PROOF By (4.54), $x^j \in V_0$ for $j = 0, 1, \dots, N - 1$. Since $w \in W_0$, $w \perp x^j$ for $j = 0, 1, \dots, N - 1$. Thus,

$$\begin{aligned} 0 &= \int_{-\infty}^{\infty} x^j w(x) dx \\ &= \int_{-\infty}^{\infty} \sum_k \left(\frac{2x - k + k}{2} \right)^j b_k \varphi(2x - k) dx \\ &= 2^{-j-1} \sum_{r=0}^j \binom{j}{r} \sum_{k=-\infty}^{\infty} k^{j-r} b_k \int_{-\infty}^{\infty} (2x - k)^r \varphi(2x - k) 2 dx \\ &= 2^{-j-1} \sum_{r=0}^j \binom{j}{r} \sum_{k=-\infty}^{\infty} k^{j-r} b_k \int_{-\infty}^{\infty} x^r \varphi(x) dx \end{aligned}$$

But

$$\int_{-\infty}^{\infty} x^r \varphi(x) dx \neq 0, \quad (4.59)$$

because of (4.50) and since x^r can be written as a linear combination of translates of φ for $r = 0, 1, \dots, N - 1$. (Recall that $x^r \in V_0$; you will fill in

the details in Exercise 39.) Thus, letting $j = 0$,

$$\sum_{k=-\infty}^{\infty} b_k = 0.$$

Letting $j = 1$,

$$\sum_{k=-\infty}^{\infty} b_k k = 0.$$

Examining the preceding computations in this proof, we can thus conclude by induction that

$$\sum_k b_k k^j = 0$$

for $j = 0, 1, \dots, N - 1$. But $b_k = (-1)^k a_{1-k}$ by Proposition 4.11. □

4.7.3 Construction of Scaling Functions

We now consider construction of

$$\varphi(x) = \sum_{\ell=0}^L a_{\ell} \varphi(2x - \ell).$$

First, observe that $a_{\ell} \neq 0$ only for $\ell = 0, 1, \dots, L$. Thus, we have $L + 1$ unknowns. Keeping this in mind, Propositions 4.10 and 4.12 give

- (A) $\sum_{k=-\infty}^{\infty} (-1)^k a_k k^j = 0$ for $j = 0, 1, \dots, N - 1$, N conditions;
- (B) $\frac{1}{2} \sum_{m=-\infty}^{\infty} a_{2k+m} a_{2\ell+m} = \delta_{k\ell}$ for $-\infty < k, \ell < \infty$, $(L + 1)/2$ conditions,
- considering the nonzero a 's and redundantly listed conditions.

Thus, we have a total of $N + (L + 1)/2$ conditions. Equating the number of variables and number of conditions shows that, given N , we need $L = 2N - 1$.

Example 4.26

$L = N = 1$.

(A) $a_0 - a_1 = 0$. Thus, $a_0 = 1$ and $a_1 = 1$.

(B) $\frac{1}{2}a_0^2 + \frac{1}{2}a_1^2 = 1$

Hence, $\varphi(x) = \varphi(2x) + \varphi(2x - 1)$, with $\varphi(x)$ supported on $[0, 1]$. (Also, $w(x) = \varphi(2x) - \varphi(2x - 1)$.) But $\varphi(x) = \varphi(2x) + \varphi(2x - 1)$ implies that φ is constant. Thus, $\varphi(x) = 1$ for $x \in [0, 1)$. Hence, V_0 consists merely of translates of $\varphi(x) = 1$ on $x \in [0, 1)$ to $\mathbb{R}$.

Consider

$$f(x) = \begin{cases} e^x, & 0 < x < 1 \\ 0, & \text{otherwise.} \end{cases}$$

Then $f \in L^2(\mathbb{R})$, and

$$f_0(x) = \sum_{k=-\infty}^{\infty} \alpha_k \varphi(x - k) = (e^1 - e^0) \varphi(x), \quad \text{and} \quad f_0(x) \in V_0,$$

$$f_{-1}(x) = \sum_{k=-\infty}^{\infty} \alpha_k \varphi(2x - k) = \begin{cases} 2(e^{1/2} - 1), & 0 < x < 1/2, \\ 2(e^1 - e^{1/2}), & 0 < x < 1, \end{cases}$$

and $f_{-1}(x) \in V_{-1}$. Here, $f_0 \in V_0$ and $f_{-1} \in V_{-1}$ are approximations to $f \in L^2(\mathbb{R})$. $\square$

Example 4.27

$N = 2$ and $L = 3$.

We need to find $\varphi(x) = a_0 \varphi(2x) + a_1 \varphi(2x - 1) + a_2 \varphi(2x - 2) + a_3 \varphi(2x - 3)$.

$$(A) \quad \begin{aligned} a_0 - a_1 + a_2 - a_3 &= 0, \\ -a_1 + 2a_2 - 3a_3 &= 0, \end{aligned}$$

$$(B) \quad \begin{aligned} \frac{1}{2}(a_0^2 + a_1^2 + a_2^2 + a_3^2) &= 1, \\ \frac{1}{2}(a_0 a_2 + a_1 a_3) &= 0. \end{aligned}$$

Solving (A) and (B) gives:

$$\begin{aligned} \varphi(x) = \frac{1}{4} \big[& (1 + \sqrt{3}) \varphi(2x) + (3 + \sqrt{3}) \varphi(2x - 1) \\ & + (3 - \sqrt{3}) \varphi(2x - 2) + (1 - \sqrt{3}) \varphi(2x - 3) \big]. \end{aligned}$$

An iterative approach is employed to find $\varphi(x)$. (The above equation only determines $\varphi(x)$ to within a constant factor.) In the iterative approach, $\varphi(x)$ is first found at $x = 1$ and $x = 2$. Specifically, since

$$\varphi(x) = \sum_{k=0}^3 a_k \varphi(2j - k) \quad \text{for } 0 \leq x < 3$$

and $\varphi(x)$ has support $[0, 3)$,

$$\varphi(j) = \sum_{k=0}^3 a_k \varphi(2j - k)$$

is a linear equation in the unknowns $\varphi(j)$, $j = 1, 2$. Also, φ is normalized so that

$$\sum_{j=1}^{2N-2} \varphi(j) = 1.$$

This determines $\varphi(j)$ for $j = 1, 2, \dots, N-2$. Values at half-integers, quarter-integers, etc. then can be determined. For example,

$$\varphi(M/2) = \sum_{k=0}^{2N-1} a_k \varphi\left(\frac{2M}{2} - k\right) = \sum_{k=0}^{2N-1} a_k \varphi(M - k).$$

The graph of such a φ (called a *Daubechies scaling function*) is illustrated in Figure 4.13. $\square$

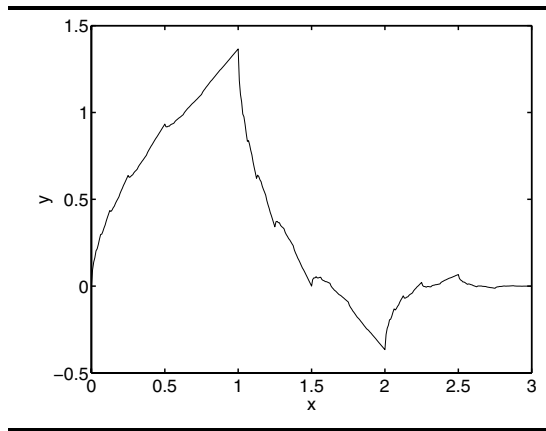


FIGURE 4.13: Illustration of a Daubechies scaling function $\varphi(x)$.

REMARK 4.60 Using several scaling functions simultaneously, wavelets that are piecewise polynomials can be constructed. (See [3, pp. 197–200].) $\square$

In §3.3.8.4 (page 139), we considered least squares approximation from a finite point set in the context of linear algebra. We now revisit this subject within the applied and approximation theoretic context of this chapter.

4.8 Least Squares Approximation on a Finite Point Set

In an earlier section of this chapter, we considered least squares approximation to $f \in C[a, b]$. For example, we find $p_n \in P^n$ such that $\|f - p_n\|_2 \leq \|f - q_n\|_2$ for all $q_n \in P^n$. We now consider least squares approximation on a finite point set.

Problem 4.1 Let G be a subset of $\mathbb{R}$ and $x_i \in G$ for $i = 1, 2, \dots, N$. Given N pairs of real numbers $(x_1, y_1), \dots, (x_N, y_N)$ and $n \leq N$ functions $u_1, u_2, \dots, u_n$ defined on G , we seek a function of the form,

$$u(x) = \sum_{k=1}^n \lambda_k u_k(x),$$

which will approximate the values $y_1, y_2, \dots, y_N$ at the points $x_1, x_2, \dots, x_N$. For example, if $u_1(x) = 1$, $u_2(x) = x$, then $u(x) = \lambda_1 + \lambda_2 x$ is a line, as is illustrated in Figure 4.14. (Note that the x_i need not be distinct.)

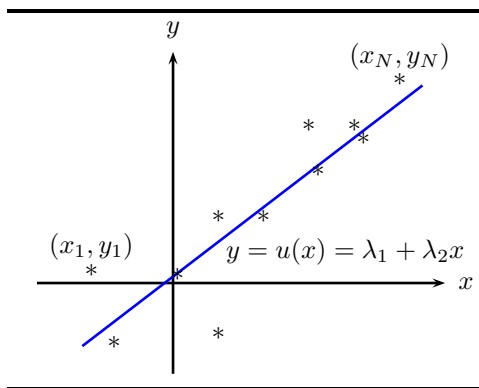


FIGURE 4.14: Graph of a possible linear least squares fit $u(x) = \lambda_1 + \lambda_2 x$.

Problem 4.1 is often called the *linear least squares problem*: Even though the fitting function $u = \sum \lambda_k u_k$ may be nonlinear, u is a linear combination of the λ_k , and the λ_k can therefore be obtained with techniques from linear algebra.

REMARK 4.61 We can extend the problem to G , a subset of $\mathbb{R}^m$ (or $\mathbb{C}^m$), where $x_i \in G$ for $i = 1, 2, \dots, N$. We then seek $u(x) = \sum_{k=1}^n \lambda_k u_k(x)$ for $x \in G$. □

REMARK 4.62 The functions $u_1, u_2, \dots, u_n$ considered in this section are powers of x , i.e., $u_i(x) = x^{i-1}$, $i = 1, 2, \dots, n$. However, other choices, such as $u_1(x) = 1$, $u_{2\ell}(x) = \sin(\ell x)$, $u_{2\ell+1}(x) = \cos(\ell x)$, $\ell = 1, 2, \dots, m$ and $n = 2m + 1$, are often used. The approximation in this case is referred to as a discrete harmonic approximation. $\square$

Least Squares Solution The least squares solution to the problem is to find $\lambda_1, \lambda_2, \dots, \lambda_n$ so that the sum

$$\sum_{j=1}^N \left(y_j - \sum_{k=1}^n \lambda_k u_k(x_j) \right)^2 \quad (4.60)$$

is minimized.

REMARK 4.63 If $n = 2$, $u_1(x) = 1$, $u_2(x) = x$, the problem reduces to the familiar problem of fitting a line to the data points, i.e., finding λ_1 and λ_2 such that $\sum_{j=1}^n (y_j - (\lambda_1 + \lambda_2 x_j))^2$ is minimized. $\square$

Now, let $y = (y_1, y_2, \dots, y_N)^T \in \mathbb{R}^N$ and

$$u_k = (u_k(x_1), u_k(x_2), \dots, u_k(x_N))^T \in \mathbb{R}^N$$

for $k = 1, 2, \dots, n$. Then (4.60) becomes the problem of minimizing $z^T z$, where $z = y - w$ and $w = \sum_{k=1}^n \lambda_k u_k$. Thus, we identify problem 4.1 with the following problem: Given vector space $V = \mathbb{R}^N$ and a subspace W of V defined by $W = \text{span}(u_1, u_2, \dots, u_n)$, find $w \in W$ such that

$$\|y - w\|^2 = (y - w)^T (y - w) \leq \|y - u\|^2$$

for all $u \in W$. Thus, w is just the best approximation to y in inner product space $\mathbb{R}^N$, and the λ_k satisfy the linear system:

$$\sum_{k=1}^n \lambda_k (u_k, u_j) = (y, u_j) \quad \text{for } j = 1, 2, \dots, n \quad (4.61)$$

Hence, $w = \sum_{k=1}^n \lambda_k u_k$ approximates y , and $w(x) = \sum_{k=1}^n \lambda_k u_k(x)$ is the solution to the least squares problem.

To find the λ_k with (4.61), it is assumed that the u_k , $1 \leq k \leq n$, are linearly independent vectors in $\mathbb{R}^N$. Thus, the functions $u_1(x), u_2(x), \dots, u_n(x)$ must be linearly independent on the subset $\{x_1, x_2, \dots, x_N\}$ of G .

Equations (4.61) can be put into the form

$$A^T A \lambda = A^T y, \quad (4.62)$$

where $A = (u_1, u_2, \dots, u_n)$ is a $N \times n$ matrix. ($A^T A$ is $n \times n$.) Note that this is precisely the form (3.29) on page 140, where we derived it from the perspective of an overdetermined linear system of equations.

REMARK 4.64 $A^T A$ is nonsingular (in fact, positive definite) if and only if $u_1, u_2, \dots, u_n$ are linearly independent. To see this, consider

$$x^T A^T A x = \sum_{i,j=1}^n (x_i u_i)(x_j u_j) = \left(\sum_{i=1}^n x_i u_i, \sum_{j=1}^n x_j u_j \right) \geq 0.$$

$A^T A$ is singular if and only if $x^T A^T A x = 0$ for some $x \neq 0$. But $x^T A^T A x = 0$ if and only if $\sum_{i=1}^n x_i u_i = 0$. However, for $x \neq 0$, $\sum_{i=1}^n x_i u_i = 0$ if and only if $u_1, u_2, \dots, u_n$ are linearly dependent. Thus, $A^T A$ is singular if and only if $u_1, u_2, \dots, u_n$ are linearly dependent. $\square$

The above discussion is summarized in the following theorem.

THEOREM 4.22

Suppose G is a subset of $\mathbb{R}$ and $x_i \in G$ for $i = 1, 2, \dots, N$. Given N pairs of real numbers $(x_1, y_1), \dots, (x_N, y_N)$, if $u_1(x), u_2(x), \dots, u_n(x)$ are linearly independent on $\{x_1, x_2, \dots, x_N\}$, then the least squares solution

$$w(x) = \sum_{k=1}^n \lambda_k u_k(x)$$

which minimizes

$$\sum_{j=1}^N (y_j - w(x_j))^2$$

is unique and is given by the solution to $A^T A \lambda = A^T y$, where

$$\begin{aligned} A &= (u_1, u_2 \dots u_n), \\ \lambda &= (\lambda_1, \lambda_2, \dots, \lambda_n)^T, \\ y &= (y_1, y_2, \dots, y_N)^T, \end{aligned}$$

and

$$u_k = (u_k(x_1), \dots, u_k(x_N))^T.$$

(Note that A is $N \times n$ and $A^T A$ is $n \times n$.)

REMARK 4.65 The so-called *smoothing polynomials* are obtained by choosing $u_i(x) = x^{i-1}$ for $i = 1, 2, \dots, n$. $\square$

By Theorem 4.22, if $u_1(x), u_2(x), \dots, u_n(x)$ are linearly independent on $\{x_1, x_2, \dots, x_N\} \subset G$, then the least squares solution exists and is unique. We have

PROPOSITION 4.13

Suppose that at least $n \leq N$ of the points $x_1, x_2, \dots, x_N$ are pairwise distinct. Then $u_1, u_2, \dots, u_n$ are linearly independent on $G \subset \mathbb{R}$, where $u_i = x^{i-1}$, $i = 1, 2, \dots, n$. (Thus, the least squares solution exists and is unique by Theorem 4.22.)

PROOF Let $\{x_1, x_2, \dots, x_n\} \subset \{x_1, x_2, \dots, x_N\}$ be n distinct points on G . Consider

$$c_1 u_1(x_i) + c_2 u_2(x_i) + \dots + c_n u_n(x_i) = 0$$

for $i = 1, 2, \dots, n$. We need to show that $c_1 = c_2 = \dots = c_n = 0$ or equivalently that $c = 0$, where $Bc = 0$ and

$$Bc = \begin{pmatrix} u_1(x_1) & u_2(x_1) & \cdots & u_n(x_1) \\ \vdots & \vdots & & \vdots \\ u_1(x_n) & u_2(x_n) & \cdots & u_n(x_n) \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

But

$$\det B = \det \begin{pmatrix} 1 & x_1 & \cdots & x_1^{n-1} \\ 1 & x_2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^{n-1} \end{pmatrix} = \prod_{\substack{j,k=1 \\ j < k}}^n (x_k - x_j) \neq 0,$$

since $x_j \neq x_k$ for $j \neq k$. Thus, $c = 0$, and $u_1, \dots, u_n$ are linearly independent. $\square$

Suppose now that the set $\{x_1, x_2, \dots, x_N\}$ has at least n distinct values. Then

$$A = \begin{pmatrix} 1 & x_1 & \cdots & x_1^{n-1} \\ 1 & x_2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_N & \cdots & x_N^{n-1} \end{pmatrix}$$

and $A^T A \lambda = A^T y$ has the form

$$\begin{pmatrix} N & \sum_{i=1}^N x_i & \sum_{i=1}^N x_i^2 & \cdots & \sum_{i=1}^N x_i^{n-1} \\ \sum_{i=1}^N x_i & \sum_{i=1}^N x_i^2 & \cdots & \sum_{i=1}^N x_i^n & \\ \vdots & & & \vdots & \\ \sum_{i=1}^N x_i^{n-1} & \cdots & \sum_{i=1}^N x_i^{2n-2} & & \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^N y_i \\ \sum_{i=1}^N x_i y_i \\ \vdots \\ \sum_{i=1}^N x_i^{n-1} y_i \end{pmatrix},$$

where $w(x) = \sum_{i=1}^n \lambda_i x^{i-1}$ is the smoothing polynomial that passes near the points $(x_1, y_1), \dots, (x_N, y_N)$.

Example 4.28

Find the least squares parabolic fit $w(x) = \lambda_1 + \lambda_2 x + \lambda_3 x^2$ for the points $(0,0), (1,2), (2,3), (3,10)$.

Solution: For this example, $N = 4$ and $n = 3$. We have the table:

i	x_i	y_i	x_i^2	x_i^3	x_i^4	$x_i y_i$	$x_i^2 y_i$
1	0	0	0	0	0	0	0
2	1	2	1	1	1	2	2
3	2	3	4	8	16	6	12
4	3	10	9	27	81	30	90
sums	6	15	14	36	98	38	104

Thus, $A^T A \lambda = A^T y$ has the form

$$\begin{pmatrix} 4 & 6 & 14 \\ 6 & 14 & 36 \\ 14 & 36 & 98 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} 15 \\ 38 \\ 104 \end{pmatrix}.$$

Solving, $\lambda_1 = 0.35$, $\lambda_2 = -0.65$, $\lambda_3 = 1.25$, and

$$w(x) = 0.35 - 0.65x + 1.25x^2$$

is the least squares parabolic fit to the points. □

REMARK 4.66 The normal equations, as in Example 4.28, are used mainly for theoretical purposes, illustration, and hand computations. Modern software uses techniques such as the QR -decomposition (§3.3.8 of this book), because $A^T A$ is ill-conditioned. Computing the QR decomposition (or the singular value decomposition of §3.5) is a more stable computation than solving the system of normal equations. □

4.9 Exercises

1. For $f(t) = \sin(t)$,
 - (a) compute the coefficients of the degree 3 Taylor polynomial approximation at $t = 0$;
 - (b) compute the coefficients of the polynomial $p_3(t)$ that minimizes

$$\int_{t=-1}^1 (f(t) - p_3(t))^2 dt;$$

- (c) compute the coefficients of the degree 3 polynomial that interpolates f at $t = -1$, $t = -1/3$, $t = 1/3$, and $t = 1$.
 - (d) Rewrite each of the degree-3 polynomials in 1a, 1b, and 1c in terms of the basis $\varphi_0 \equiv 1$, $\varphi_1 \equiv t$, $\varphi_2 \equiv t^2$, and $\varphi_3 \equiv t^3$, then compare coefficients.
 - (e) Estimate the infinity norm of the error, for each of the approximations in 1a, 1b, and 1c.
2. Find the least squares approximation for the function $f(x) = e^x$ by a polynomial $p_1(x) = b_0 + b_1x$ on $[-1, 1]$ using the norm

$$\|f\|^2 = \int_{-1}^1 (f(x))^2 dx.$$

3. Let $f \in C[-1, 3]$ and let $\mathcal{P}^n = \text{span}\{\varphi_0, \varphi_1, \dots, \varphi_n\}$, with $\{\varphi_i\}_{i=0}^n$ orthonormal. Let $g^* \in \mathcal{P}^n$ be the least squares approximation to f , i.e., for any $p \in \mathcal{P}^n$, $\|g^* - f\|^2 \leq \|p - f\|^2$, where $\|f\|^2 = \int_{-1}^3 (f(x))^2 dx$. Prove that $\int_{-1}^3 (f(x) - g^*(x))p(x)dx = 0$ for all $p \in \mathcal{P}^n$.
4. Use Gram-Schmidt orthogonalization process to compute the first three polynomials $q_0(x)$, $q_1(x)$, and $q_2(x)$ which are orthogonal on the interval $[0, 1]$ with respect to the weight function $w(x) = 1$. Using these polynomials, obtain the least squares approximation of second degree for $f(x) = x^{1/2}$ on $[0, 1]$.
5. Consider *Runge's function*: $f(x) = \frac{1}{1 + x^2}$.
 - (a) Compute and graph the interpolating polynomials (with a graph of Runge's function itself) using 5, 9, and 17 equally spaced points in the interval $[-5, 5]$. What do you find?

- (b) Use the error formula for polynomial interpolation to estimate the error in each of the interpolants in part 5a.
 - (c) Repeat part 5a, except using Chebyshev points instead of equally spaced points.
 - (d) Use the error formula for polynomial interpolation to estimate the error in each of the interpolants in part 5c.
6. Let P_1 be the a polynomial of linear degree (degree 1) interpolating $f(x)$ at the points x_0, x_1 .
- a) Show that P_1 is unique. Moreover, if $f \in C^2[x_0, x_1]$, show that the error in the linear interpolant satisfies

$$f(x) - P_1(x) = \frac{1}{2}(x - x_0)(x - x_1)f''(\xi(x)), \quad x_0 < x < x_1.$$

- b) Find the function $\xi(x)$ explicitly for $f(x) = 1/x$, $x_0 = 1$, and $x_1 = 2$. Furthermore, find $\max_{1 \leq x \leq 2} \xi(x)$ and $\min_{1 \leq x \leq 2} \xi(x)$.

7. Show that, in the setting of Remark 4.15 (on page 218),

$$\|W\|_\infty = 2^{-2n-1}(b-a)^{n+1}.$$

8. Show that the matrix A in the system (4.21) on page 222 is positive definite.
9. (Literature search question) What is the constant c_k in Remark 4.27 on page 233? What happens if $f \in C^\infty[a, b]$?
10. Repeat the computations for Example 4.10 on page 237, except use the interpolating polynomial at the six Chebyshev points, rather than at six equally spaced points.
11. Let $f \in C[0, 1]$ and let $\sum_{i=1}^n w_i f(x_i)$ be an approximation to $\int_0^1 f(x) dx$. (The x_i are called points and the w_i are called weights.) Assume that $0 \leq w_i \leq 1$ and $0 \leq x_i \leq 1$ for $i = 1, 2, \dots, n$, and assume $\sum_{i=1}^n w_i = 1$ for any n . Let

$$E_n(f) = \int_0^1 f(x) dx - \sum_{i=1}^n w_i f(x_i)$$

be the error in the approximation. Assume that $E_n(P_n) = 0$ for P_n any polynomial of degree $\leq n$. Use the Weierstrass approximation theorem to prove that, given $\epsilon > 0$, there is an $N > 0$ such that $|E_n(f)| < \epsilon$ when $n > N$.

12. Let $f \in C^\infty[0, 1]$ be such that

$$\max_{0 \leq x \leq 1} |f^{(n+1)}(x)| \leq K(n+2)! \quad \text{for } n = 1, 2, \dots \text{ and some } K > 0.$$

Let $x_i = i/n$, $i = 0, 1, 2, \dots, n$, and let $P_n(x)$ interpolate $f(x)$ at $x = x_i$ for $i = 0, 1, 2, \dots, n$. Prove that, given $\epsilon > 0$, there is an $N > 0$ such that $\|f - P_n\|_\infty < \epsilon$ when $n \geq N$.

13. The polynomial $P_n(x)$ interpolating the function $f(x)$ at the nodes x_k for $k = 0, \dots, n$ is given by

$$P_n(x) = \sum_{k=0}^n L_k(x) f(x_k), \quad \text{where} \quad L_k(x) = \prod_{i=0, i \neq k}^n \frac{(x - x_i)}{(x_k - x_i)}.$$

Let $\psi(x) = \prod_{i=0}^n (x - x_i)$. Show that $\sum_{k=0}^n L_k(x) = 1$ and

$$P_n(x) = \psi(x) \sum_{k=0}^n \frac{f(x_k)}{(x - x_k)\psi'(x_k)}.$$

14. Consider the function $P_3(x) = x^3 - 9x^2 - 20x + 5$. Find a second degree polynomial $P_2(x)$ such that $\delta = \max_{0 \leq x \leq 4} |P_3(x) - P_2(x)|$ is as small as possible.

15. Consider interpolating the function $f(x, y)$ at the n^2 points (x_i, y_j) for $i, j = 1, 2, \dots, n$, where $\{x_i\}_{i=1}^n$ and $\{y_j\}_{j=1}^n$ are each pairwise distinct. Let

$$l_i(x) = \prod_{\substack{m=1 \\ m \neq i}}^n \frac{x - x_m}{x_i - x_m}, \quad \hat{l}_j(y) = \prod_{\substack{k=1 \\ k \neq j}}^n \frac{y - y_k}{y_j - y_k},$$

$$\text{and } p(x, y) = \sum_{i=1}^n \sum_{j=1}^n c_{ij} l_i(x) \hat{l}_j(y).$$

- (a) Find c_{ij} for $i, j = 1, \dots, n$ so that $p(x, y)$ interpolates $f(x, y)$ at the n^2 points.

- (b) Show that $\sum_{i=1}^n \sum_{j=1}^n l_i(x) \hat{l}_j(y) = 1$.

16. Suppose the $N_k(x)$ are as in (4.10) on page 215, and form the system of equations in the unknowns c_k associated with the conditions

$$\{p(x_i) = y_i\}_{i=0}^n$$

with

$$p(x) = \sum_{k=0}^n c_k N_k(x),$$

analogously to how we formed the Vandermonde system and the (trivial) system of equations associated with the Lagrange basis functions.

- (a) Show that the matrix for this system of equations is lower triangular.
 - (b) What are the c_k ?
17. Let $x_0, x_1, \dots, x_n$ be distinct real numbers, and consider the interpolation problem $P_n(x) = \sum_{k=0}^n c_k e^{kx}$ such that $P_n(x_i) = y_i$, $i = 0, 1, 2, \dots, n$, with $\{y_i\}_{i=0}^n$ arbitrary real values. Prove that there is a unique choice for the coefficients $c_0, c_1, c_2, \dots, c_n$.
 18. Using Hermite interpolation, find the polynomial $P_2(x)$ that satisfies $P_2(0) = f(0)$, $P_2(2) = f(2)$ and $P_2'(2) = f'(2)$. Also estimate the error $|f(x) - P_2(x)|$ for $f \in C^{(3)}[0, 1]$.
 19. Let $x_i = ih$ for $i = 0, 1, \dots, N$ be $N + 1$ distinct points on $[0, 1]$, where $h = 1/N$. Assume that $f \in C^4[0, 1]$. Let $\hat{H}_2(x)$ be the piecewise cubic Hermite interpolant to $f(x)$ such that $\hat{H}_2(x_i) = f(x_i)$ and $\hat{H}_2'(x_i) = f'(x_i)$ for $i = 0, 1, \dots, N$. Prove that

$$\max_{0 \leq x \leq 1} |f(x) - \hat{H}_2(x)| \leq \frac{h^4}{16} \frac{\|f^{(4)}\|_\infty}{4!}.$$

20. Find the best uniform approximation (minimax approximation) $a_0 + a_1x$ to $f(x) = \ln(1 + x)$ on the interval $[0, 1]$. Note that

$$\max_{0 \leq x \leq 1} |\ln(1 + x) - a_0 - a_1x| \leq \max_{0 \leq x \leq 1} |\ln(1 + x) - b_0 - b_1x|$$

for any $b_0, b_1 \in \mathbb{R}$.

21. Suppose that $f(x)$ is an even function on $[-a, a]$. Show that the best uniform approximation $P_n^*(x)$ to $f(x)$ on $[-a, a]$ is also even.
22. Complete the computations in Example 4.10 on page 237. That is, approximate $\sin(x)$, $x \in [-0.1, 0.1]$ by an interpolating polynomial of degree 5, and use interval arithmetic to obtain an interval bounding the exact solution of $\sin(0.05)$. Do it

- (a) with equally spaced points, and
- (b) with Chebyshev points.

23. Prove Remark 4.36 (on page 243).

24. Show that A is as in (4.30) on page 244 (in the context of the proof of Theorem 4.14).
25. Show that conditions (n) in Definition 4.17 of the natural spline interpolant (on page 246) lead to the system (4.34) (on page 247).
26. Let $s_1(x) = 1 + c(x+1)^3$, $-1 \leq x \leq 0$, where c is a real number. Determine $s_2(x)$ on $0 \leq x \leq 1$ so that

$$s(x) = \begin{cases} s_1(x), & -1 \leq x \leq 0, \\ s_2(x), & 0 \leq x \leq 1, \end{cases}$$

is a natural cubic spline, i.e. $s''(-1) = s''(1) = 0$ on $[-1, 1]$ with nodal points at $-1, 0, 1$. How must c be chosen if one wants $s(1) = -1$?

27. Suppose that $f(x)$ satisfies a Lipschitz condition $|f(x) - f(y)| \leq L|x - y|$ for all $x, y \in [0, 1]$. Let $\Psi(x)$ be a piecewise constant approximation to $f(x)$ such that $\Psi(x) = f(x_i)$ for $x_i \leq x < x_{i+1}$, for $i = 0, 1, \dots, N-1$, with $x_i = ih$ and $h = 1/N$. Prove that $\max_{0 \leq x \leq 1} |\Psi(x) - f(x)| \leq ch$ for some constant $c > 0$.

28. Let $a = t_0 < t_1 < \dots < t_n = b$. We wish to determine a function $S \in C^1[a, b]$ such that

(i) $S(t_i) = f(t_i)$, $i = 0, 1, \dots, n$.

(ii) $S(t) = S_j(t)$ for $t \in [t_{j-1}, t_j]$, where S_j is a quadratic polynomial.

(a) With $h_j = t_j - t_{j-1}$, show that

$$S'(t_{j-1}) + S'(t_j) = \frac{2}{h_j} (f(t_j) - f(t_{j-1})), \quad j = 1, \dots, n.$$

(b) If $S'(t_0) = f'(t_0)$, find the linear system $S'(t_1), \dots, S'(t_n)$ must satisfy.

29. Prove Remark 4.43 on page 256.

30. Let $f(x) = (\cos x - 0.5)^{\frac{5}{2}}$. Let

$$g_k(x) = \sum_{j=-k}^k \alpha_j e^{ijx}, \quad \text{where } \alpha_j = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ijx} dx.$$

Prove that there is a constant $c > 0$ such that $\|g_k - f\|_\infty \leq c/k$.

31. Let $f \in C^{2n+1}[-\pi, \pi]$. Let $\hat{f}(x) = \sum_{k=-n}^n c_k e^{ikx}$ be a trigonometric approximation to $f(x)$ such that $f^{(l)}(0) = \hat{f}^{(l)}(0)$ for $l = 0, 1, 2, \dots, 2n$. ($\hat{f}(x)$ can be considered a Taylor trigonometric approximation.)

- (a) Show that $c_k, -n \leq k \leq n$ can be determined, so $\hat{f}(x)$ exists.
- (b) Find $\hat{f}(x)$ for $f(x) = \pi/(x + 3\pi)$ and $n = 1$. That is, find coefficients c_{-1}, c_0, c_1 such that $\hat{f}(x) = c_{-1}e^{-ix} + c_0 + c_1e^{ix}$. Then, write $\hat{f}(x)$ in terms of real trigonometric functions.

32. Consider

$$f(x) = \begin{cases} -1, & 0 \leq x \leq \pi, \\ 1, & \pi \leq x \leq 2\pi. \end{cases}$$

- (a) Use the FFT procedure with $N = 16$ to compute the trigonometric interpolating polynomial to f . Arrange your computations in a table similar to the table on page 258.
- (b) Graph the trigonometric interpolating polynomial you have obtained. (You can use MATLAB, for example, to make the graph.)
33. Let $f(x) = \ln(1 + x)$ (as in Example 4.15 on page 264) for $x \geq 0$, and consider possible interpolation at the points $x_1 = 0, x_2 = .05, x_3 = 0.1, x_4 = 0.15$ with an $L = 2, M = 1$ rational function.

- (a) If it is not possible to so interpolate f , then explain why.
- (b) If it is possible to so interpolate f , then find the coefficients.
- (c) If it is possible to so interpolate f , then use (4.47) (on page 266) to compute a bound on the interpolation error for $0 \leq x \leq 0.15$, then compare this bound to the error bound obtained for the Padé approximant in Example 4.15.

(Hint: Let $p(x) = a_0 + a_1x + a_2x^2$ and $q(x) = b_0 + b_1x$, then consider the homogeneous system of equations (4.48) with the additional condition $b_0 = 1$.)

34. Consider the function $f(x) = \int_0^x e^{-s^2} ds$.

- (a) Find the $R(1, 2)$ Padé approximant of $f(x)$.
- (b) Estimate the maximum error in the above approximation for $0 \leq x \leq 2$.

35. Obtain the $R(1, 2)$ rational approximation to the function e^x of the form

$$\frac{a_0 + a_1x}{1 + b_1x + b_2x^2}.$$

36. Show that the $R(0, 1)$ Padé approximant to $f(x) = x$ does not exist.

37. Find the $R(0, 1)$ Padé approximant to $f(x) = 2 + 3x$.

38. Prove the assertions in Example 4.19 (on page 268). That is, using (4.49) and (4.50), prove that, for $L = 1$, $a_0 = 1$, and $a_2 = 1$, that
- (a) φ is supported on $[0, 1)$, and
 - (b) φ is constant on $[0, 1)$.
39. Fill in the details of why (4.59) on page 275 is true.

Chapter 5

Eigenvalue-Eigenvector Computation

This chapter is concerned with the computation of eigenvalues and eigenvectors. Several good references for the material in this chapter are [49], [85], [97], and [101]. Before studying numerical methods for accomplishing this task, a few important definitions and results from matrix theory are presented (many without proofs).

Computing the eigenvalues and eigenvectors of a matrix is inherently an iterative computational problem. In particular, Niels Abel proved that quintic equations are insoluble by finite algebraic formulae. Since the eigenvalue problem can be formulated as solution of an algebraic equation, i.e., $\det(A - \lambda I) = 0$, Abel's result implies that eigenvalue computation requires iterative numerical methods of solution for $n \times n$ matrices with $n > 4$.

5.1 Basic Results from Linear Algebra

DEFINITION 5.1 Let A be an $n \times n$ complex matrix and $x \in \mathbb{C}^n$. Then x is an eigenvector of A corresponding to eigenvalue λ if $x \neq 0$ and $Ax = \lambda x$. (A vector y such that $y^H A = \lambda y^H$ is called a left eigenvector, and, in general, $x \neq y$.)

REMARK 5.1 λ is an eigenvalue of A if and only if $\det(A - \lambda I) = 0$. The determinant defines the *characteristic polynomial*

$$\det(A - \lambda I) = \lambda^n + \alpha_{n-1}\lambda^{n-1} + \alpha_{n-2}\lambda^{n-2} + \cdots + \alpha_1\lambda + \alpha_0.$$

Thus, A has exactly n eigenvalues, the roots of the above polynomial, in the complex plane, counting multiplicities. The set of eigenvalues of A is called the *spectrum* of A . Recall the following.

- (a) The *spectral radius* of A is defined by

$$\rho(A) = \max_{\lambda \text{ an eigenvalue of } A} |\lambda|.$$

- (b) $|\lambda| \leq \|A\|$ for any induced matrix norm $\|\cdot\|$ and any eigenvalue λ .
- (c) $\|A\|_2 = \sqrt{\rho(A^H A)}$. If $A^H = A$ (that is, if A is Hermitian), then $\|A\|_2 = \rho(A)$.

□

DEFINITION 5.2 A square matrix A is called defective if it has an eigenvalue of multiplicity k having fewer than k linearly independent eigenvectors.

For example, if

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \text{ then } \lambda_1 = \lambda_2 = 1, \text{ but } x = c \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

is the only eigenvector, so A is defective.

PROPOSITION 5.1

Let A and P be $n \times n$ matrices, with P nonsingular. Then λ is an eigenvalue of A with eigenvector x if and only if λ is an eigenvalue of $P^{-1}AP$ with eigenvector $P^{-1}x$. ($P^{-1}AP$ is called a similarity transformation of A , and A and $P^{-1}AP$ are called similar.)

PROPOSITION 5.2

Let $\{x_i\}_{i=1}^n$ be eigenvectors of A corresponding to distinct eigenvalues $\{\lambda_i\}_{i=1}^n$. Then the vectors $\{x_i\}_{i=1}^n$ are linearly independent.

REMARK 5.2 If A has n different eigenvalues, then the n eigenvectors are linearly independent and thus form a basis for $\mathbb{C}^n$. (Note that n different eigenvalues is sufficient but not necessary for $\{x_i\}_{i=1}^n$ to form a basis. Consider $A = I$ with eigenvectors $\{e_i\}_{i=1}^n$.) □

PROPOSITION 5.3

Let A be an $n \times n$ complex matrix. Then A is nondefective (that is, A has a complete set of eigenvectors) if and only if there is a nonsingular matrix X such that $X^{-1}AX = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, where $\{\lambda_i\}_{i=1}^n$ are the eigenvalues of A . The i -th column of X is an eigenvector corresponding to λ_i and the i -th row of X^{-1} is a left eigenvector corresponding to λ_i .

PROPOSITION 5.4

(Schur decomposition) Let A be an $n \times n$ complex matrix. Then there exists a unitary matrix U ($U^H U = I$) such that $U^H A U$ is upper triangular with diagonal elements $\lambda_1, \lambda_2, \dots, \lambda_n$.

We now have the following useful result:

THEOREM 5.1

(Gerschgorin's Circle Theorem) Let A be any $n \times n$ complex matrix. Then every eigenvalue of A lies in the union of the discs

$$\bigcup_{j=1}^n K_{\rho_j}(a_{jj}), \quad \text{where } K_{\rho_j} = \{z \in \mathbb{C} : |z - a_{jj}| \leq \rho_j\} \quad \text{for } j = 1, 2, \dots, n,$$

and where the centers a_{jj} are diagonal elements of A and the radii ρ_j can be taken as:

$$\rho_j = \sum_{\substack{k=1 \\ k \neq j}}^n |a_{jk}|, \quad j = 1, 2, \dots, n \quad (5.1)$$

(absolute sum of the elements of each row excluding the diagonal elements),

$$\rho_j = \sum_{\substack{k=1 \\ k \neq j}}^n |a_{kj}|, \quad j = 1, 2, \dots, n \quad (5.2)$$

(absolute column sums excluding diagonal elements), or

$$\rho_j = \rho = \left(\sum_{\substack{j,k=1 \\ j \neq k}}^n |a_{jk}|^2 \right)^{1/2} \quad (5.3)$$

for $j = 1, 2, \dots, n$.

Example 5.1

$$A = \begin{pmatrix} 2 & 1 & \frac{1}{2} \\ -1 & -3i & 1 \\ 3 & -2 & -6 \end{pmatrix}$$

Using absolute row sums,

$$\begin{aligned} a_{11} &= 2, & \rho_1 &= 3/2, \\ a_{22} &= -3i, & \rho_2 &= 2, \\ a_{33} &= -6, & \rho_3 &= 5. \end{aligned}$$

The eigenvalues are in the union of these discs. For example, $\rho(A) \leq 11$. Also, A is nonsingular, since no eigenvalue λ can equal zero. (See Figure 5.1.) $\square$

PROOF (of Theorem 5.1) Let λ be any eigenvalue of A . We will show that λ must lie in one of the circles.

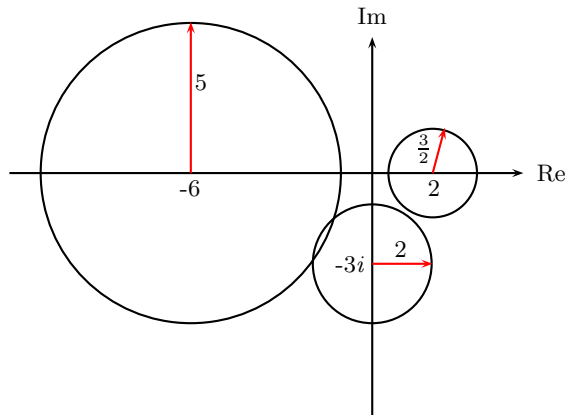


FIGURE 5.1: Illustration of Gerschgorin discs for Example 5.1.

Case 1 Suppose that $\lambda = a_{jj}$ for some j . Then the assertion is satisfied.

Case 2 Suppose that $\lambda \neq a_{jj}$ for any j , $j = 1, 2, \dots, n$.

Consider $\lambda I - D$ where D is the diagonal matrix $d_{jj} = a_{jj}$, $\lambda I - D$ is nonsingular and $(\lambda I - D)^{-1} = \text{diag}(1/(\lambda - a_{jj}))$. Let w be an eigenvector of A corresponding to eigenvalue λ . Then $Aw = \lambda w$ and thus $(A - D)w = (\lambda I - D)w$ so $w = (\lambda I - D)^{-1}(A - D)w$.

Let $\|\cdot\|$ be a matrix norm compatible with a vector norm, i.e., $\|Ax\| \leq \|A\|\|x\|$. In particular,

$$\begin{aligned}\|Ax\|_{\infty} &\leq \|A\|_{\infty}\|x\|_{\infty} \\ \|Ax\|_1 &\leq \|A\|_1\|x\|_1 \\ \|Ax\|_2 &\leq \|A\|_2\|x\|_2, \text{ or} \\ \|Ax\|_2 &\leq \|A\|_E\|x\|_2.\end{aligned}$$

Then,

$$\|w\| = \|(\lambda I - D)^{-1}(A - D)w\| \leq \|(\lambda I - D)^{-1}(A - D)\|\|w\|.$$

Hence, $1 \leq \|(\lambda I - D)^{-1}(A - D)\|$.

We now consider $\|\cdot\| = \|\cdot\|_{\infty}$, $\|\cdot\|_1$, and $\|\cdot\|_E$.

(A) $\|\cdot\| = \|\cdot\|_{\infty}$ We have:

$$1 \leq \max_{1 \leq j \leq n} \sum_{\substack{k=1 \\ k \neq j}}^n \frac{|a_{jk}|}{|\lambda - a_{jj}|} = \max_{1 \leq j \leq n} \frac{\rho_j}{|\lambda - a_{jj}|} \quad \text{using Eq. (5.1) for } \rho_j.$$

Thus, $|\lambda - a_{jj}| \leq \rho_j$ for at least one j . Hence, $\lambda \in K_{\rho_j}(a_{jj})$ for some j .

(B) $\|\cdot\| = \|\cdot\|_1$ We have:

$$1 \leq \max_{1 \leq j \leq n} \sum_{\substack{k=1 \\ k \neq j}}^n \frac{|a_{kj}|}{|\lambda - a_{jj}|} = \max_{1 \leq j \leq n} \frac{\rho_j}{|\lambda - a_{jj}|} \quad \text{using Eq. (5.2) for } \rho_j.$$

Thus, $|\lambda - a_{jj}| \leq \rho_j$ for at least one j . Hence, $\lambda \in K_{\rho_j}(a_{jj})$ for some j .

(C) $\|\cdot\| = \|\cdot\|_E$ We have:

$$1 \leq \left(\sum_{\substack{j,k=1 \\ j \neq k}}^n \frac{|a_{jk}|^2}{|\lambda - a_{jj}|^2} \right)^{\frac{1}{2}} \leq \frac{\rho}{\min_{1 \leq j \leq n} |\lambda - a_{jj}|} \quad \text{using Eq. (5.3) for } \rho.$$

Thus, $|\lambda - a_{jj}| \leq \rho$ for at least one j . Hence, $\lambda \in K_{\rho_j}(a_{jj})$ for some j .

□

REMARK 5.3 It can be shown that if $\hat{S}$ is the union of m discs K_{ρ_j} such that $\hat{S}$ is disjoint from all the other disks, then $\hat{S}$ contains precisely m eigenvalues. (See [68].) □

We now consider some results for the special case when matrix A is Hermitian. Recall that, if $A^H = A$, then A is called Hermitian. (A real symmetric matrix is a special kind of Hermitian matrix.)

THEOREM 5.2

Let A be Hermitian (or real symmetric). The eigenvalues of A are real, and there is an orthonormal system of eigenvectors $w_1, w_2, \dots, w_n$ of A with $Aw_j = \lambda_j w_j$ and $(w_j, w_k) = w_k^H w_j = \delta_{jk}$.

REMARK 5.4 The orthonormal system is linearly independent and spans $\mathbb{C}^n$, and thus forms a basis for $\mathbb{C}^n$. Thus, any vector $x \in \mathbb{C}^n$ can be expressed as

$$x = \sum_{j=1}^n a_j w_j, \quad \text{where } a_j = (x, w_j) \quad \text{and } \|x\|_2^2 = \sum_{i=1}^n |a_i|^2.$$

□

We have two more interesting results concerning eigenvalues of Hermitian matrices.

THEOREM 5.3

Let A and B be Hermitian matrices. Then, the eigenvalues $\lambda_j(A)$, $\lambda_j(B)$, $j = 1, 2, \dots, n$ arranged in the order $\lambda_1(A) \geq \lambda_2(A) \geq \dots \geq \lambda_n(A)$, $\lambda_1(B) \geq \lambda_2(B) \geq \dots \geq \lambda_n(B)$ satisfy the inequality

$$|\lambda_j(A) - \lambda_j(B)| \leq \|A - B\|, \quad j = 1, 2, \dots, n$$

for any matrix norm compatible with a vector norm, i.e., $\|Ax\| \leq \|A\|\|x\|$.

PROOF (See [90].) □

REMARK 5.5 If the elements of A and B are close, then Theorem 5.3 asserts that the eigenvalues are close. □

We now have a modified form of Gerschgorin's Theorem for Hermitian matrices.

COROLLARY 5.1

(A Gerschgorin Circle Theorem for Hermitian matrices) If A is Hermitian or real symmetric, then a one-to-one correspondence can be set up¹ between each disc $K_\rho(a_{jj})$ and each λ_j from the spectrum $\lambda_1, \lambda_2, \dots, \lambda_n$ of A , where

$$\rho = \max_j \sum_{\substack{k=1 \\ k \neq j}}^n |a_{jk}| \quad \text{or} \quad \rho = \left(\sum_{\substack{j,k=1 \\ j \neq k}}^n |a_{jk}|^2 \right)^{1/2}.$$

(Recall that $K_\rho(a_{jj}) = \{z \in \mathbb{C} : |z - a_{jj}| \leq \rho\}$.)

PROOF Let $D = \text{diag}(a_{ii})$. The eigenvalues of D are the diagonal elements $a_{11}, a_{22}, \dots, a_{nn}$, which are real since A is Hermitian. Let $a'_{11}, \dots, a'_{nn}$ be a permutation of $a_{11}, a_{22}, \dots, a_{nn}$ so that $a'_{11} > a'_{22} > \dots > a'_{nn}$. Using Theorem 5.3, we have $|\lambda_j(A) - a'_{jj}| \leq \|A - D\|_\alpha$ for $j = 1, 2, \dots, n$ where $\alpha = E$ or ∞ . (Note that $\alpha = 1$ yields, by symmetry the same result as $\alpha = \infty$.)

Letting $\alpha = \infty$, we have

$$|\lambda_j(A) - a'_{jj}| \leq \rho = \max_j \sum_{\substack{k=1 \\ k \neq j}}^n |a_{jk}|, \quad j = 1, 2, \dots, n.$$

¹This does not necessarily mean that there is only one eigenvalue in each disk; consider the matrix $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

Letting $\alpha = E$, we have

$$|\lambda_j(A) - a'_{jj}| \leq \rho = \left(\sum_{\substack{j,k=1 \\ j \neq k}}^n |a_{jk}|^2 \right)^{1/2}, \quad j = 1, 2, \dots, n.$$

Thus, eigenvalue $\lambda_j(A)$ lies in the closed interval $K_\rho(a'_{jj}) = [a'_{jj} - \rho, a'_{jj} + \rho]$. $\square$

Example 5.2

$$A = \begin{pmatrix} 3 & -1 & 1 \\ -1 & 15 & 1 \\ 1 & 1 & 10 \end{pmatrix}.$$

We have $a'_{11} = 15$, $a'_{22} = 10$, $a'_{33} = 3$,

$$\max_j \sum_{\substack{k=1 \\ k \neq j}}^n |a_{jk}| = 2, \quad \text{and} \quad \left(\sum_{\substack{j,k=1 \\ j \neq k}}^n |a_{jk}|^2 \right)^{1/2} = 6^{1/2},$$

so we may use $\rho = 2$. Thus, $13 \leq \lambda_1 \leq 17$, $8 \leq \lambda_2 \leq 12$, $1 \leq \lambda_3 \leq 5$. $\square$

5.2 The Power Method

In this section, we describe a simple iterative method for computing the eigenvector corresponding to the largest (in modulus) eigenvalue of a matrix A . We assume first that A is nondefective, i.e., A has a complete set of eigenvectors, and A has a unique simple² dominant eigenvalue. We will discuss more general cases later.

Specifically, suppose that the $n \times n$ matrix A has a complete set of eigenvectors corresponding to eigenvalues $\{\lambda_j\}_{j=1}^n$, and the eigenvalues satisfy

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|. \quad (5.4)$$

Since the eigenvectors $\{x_j\}$ are linearly independent, they form a basis for $\mathbb{C}^n$. That is, any vector $q^{(0)} \in \mathbb{C}^n$ can be written

$$q^{(0)} = \sum_{j=1}^n c_j x_j, \quad (5.5)$$

²A simple eigenvalue is an eigenvalue corresponding to a root of multiplicity 1 of the characteristic equation

for some coefficients c_j . Starting with initial guess $q^{(0)}$, we define the sequence $\{q^{(\nu)}\}_{\nu \geq 1}$ by

$$q^{(\nu+1)} = \frac{1}{\sigma_{\nu+1}} Aq^{(\nu)}, \quad \nu = 0, 1, 2, \dots \quad (5.6)$$

where the sequence $\{\sigma_\nu\}_{\nu \geq 1}$ consists of scale factors chosen to avoid overflow and underflow errors. From (5.5) and (5.6), we have

$$\begin{aligned} q^{(\nu)} &= \left[\prod_{i=1}^{\nu} \sigma_i^{-1} \right] \sum_{j=1}^n \lambda_j^\nu c_j x_j \\ &= \lambda_1^\nu \left[\prod_{i=1}^{\nu} \sigma_i^{-1} \right] \left\{ c_1 x_1 + \sum_{j=2}^n \left(\frac{\lambda_j}{\lambda_1} \right)^\nu c_j x_j \right\}. \end{aligned} \quad (5.7)$$

Since by (5.4), $|\lambda_j/\lambda_1| < 1$ for $j \geq 2$, we have $\lim_{\nu \rightarrow \infty} (\lambda_j/\lambda_1)^\nu = 0$ for $j \geq 2$, and if $c_1 \neq 0$,

$$\lim_{\nu \rightarrow \infty} q^{(\nu)} = \lim_{\nu \rightarrow \infty} \left[\lambda_1^\nu \prod_{i=1}^{\nu} \frac{1}{\sigma_i} \right] c_1 x_1. \quad (5.8)$$

The scale factors σ_i are usually chosen so that $\|q^{(\nu)}\|_\infty = 1$ or $\|q^{(\nu)}\|_2 = 1$ for $\nu = 1, 2, 3, \dots$, i.e., the vector $q^{(\nu)}$ is normalized to have unit norm; thus $\sigma_{\nu+1} = \|Aq^{(\nu)}\|_\infty$ or $\|Aq^{(\nu)}\|_2$, since $q^{(\nu+1)} = Aq^{(\nu)}/\sigma_{\nu+1}$. With either normalization, the limit in (5.8) exists; in fact,

$$\lim_{\nu \rightarrow \infty} q^{(\nu)} = \frac{x_1}{\|x_1\|}, \quad (5.9)$$

i.e., the sequence $q^{(\nu)}$ converges if $c_1 \neq 0$ to an eigenvector of unit length corresponding to the dominant eigenvalue of A .

REMARK 5.6 If $q^{(0)}$ is chosen randomly, the probability that $c_1 \neq 0$ is close to one, but not one. However, even if the exact $q^{(0)}$ happens to have been chosen with $c_1 = 0$, rounding errors on the computer may still result in a component in direction x_1 . $\square$

Example 5.3

$$A = \begin{pmatrix} -4 & 14 & 0 \\ -5 & 13 & 0 \\ -1 & 0 & 2 \end{pmatrix}.$$

$\lambda_1 = 6$, $\lambda_2 = 3$, $\lambda_3 = 2$. (A is nondefective because all the eigenvalues are distinct.) Let $q^{(0)} = (1 \ 1 \ 1)^T$. Then,

$$q^{(1)} = \frac{1}{\sigma_1} Aq^{(0)} = \frac{1}{\sigma_1} (10 \ 8 \ 1)^T = (1 \ 0.8 \ 0.1)^T \quad (\text{choosing } \sigma_{\nu+1} = \|Aq^{(\nu)}\|_\infty).$$

Thus,

$$\begin{aligned}
 q^{(2)} &= \frac{1}{\sigma_2} Aq^{(1)} \approx (1 \quad 0.75 \quad -0.111)^T \\
 q^{(3)} &= \frac{1}{\sigma_3} Aq^{(2)} \approx (1 \quad 0.731 \quad -0.188)^T \\
 q^{(4)} &\approx (1 \quad 0.722 \quad -0.221)^T, \\
 q^{(5)} &\approx (1 \quad 0.718 \quad -0.235)^T, \\
 &\vdots \\
 q^{(10)} &\approx (1 \quad 0.714 \quad -0.250)^T, \\
 q^{(11)} &\approx (1 \quad 0.714 \quad -0.250)^T.
 \end{aligned}$$

Hence, the eigenvector corresponding to $\lambda_1 = 6$ is approximately

$$(1 \quad 0.714 \quad -0.250)^T.$$

(Of course, as described later, we also obtain an approximation to λ_1 .) $\square$

Consider again Eq. (5.7). Since by assumption λ_2 is the eigenvalue of second largest absolute magnitude, we see that, for ν sufficiently large,

$$\frac{q^{(\nu)} - \left(\lambda_1^\nu \prod_{i=1}^n \sigma_i^{-1} \right) c_1 x_1}{(\lambda_2/\lambda_1)^\nu} \rightarrow k \quad \text{as } \nu \rightarrow \infty,$$

where k is a constant vector. Hence,

$$q^{(\nu)} - \frac{x_1}{\|x_1\|} = \mathcal{O} \left(\left| \frac{\lambda_2}{\lambda_1} \right|^\nu \right). \quad (5.10)$$

That is, the rate of convergence of the sequence $q^{(\nu)}$ to the exact eigenvector is governed by the ratio $|\lambda_2/\lambda_1|$. In practice, this ratio may be too close to 1, yielding a slow convergence rate. For instance, if $|\lambda_2/\lambda_1| = 0.95$, then $|\lambda_2/\lambda_1|^\nu \leq 0.1$ only for $\nu \geq 44$, that is, it takes over 44 iterations to reduce the error in (5.10) by a factor of 10. On the other hand, the method is very simple, requiring n^2 multiplications to compute $Aq^{(\nu)}$ at each iteration. Also, if A is sparse, the work is reduced, and only the nonzero elements of A need to be stored.

Now having obtained a sequence of vectors converging to x_1 , we consider finding an approximation to λ_1 itself. There are two popular ways of doing this in conjunction with the power method. In the first approach, one computes the approximation as

$$\mu_{\nu+1} = \frac{(Aq^{(\nu)})_k}{(q^{(\nu)})_k} = \frac{\sigma_{\nu+1} q_k^{(\nu+1)}}{q_k^{(\nu)}}, \quad (5.11)$$

where $q_k^{(\nu)}$ is the k -th component of $q^{(\nu)}$, usually the largest in modulus. Substituting (5.7) into (5.11) yields

$$\mu_{\nu+1} = \lambda_1 \left\{ \frac{c_1 x_{1k} + \sum_{j=2}^n \left(\frac{\lambda_j}{\lambda_1} \right)^{\nu+1} c_j x_{jk}}{c_1 x_{1k} + \sum_{j=2}^n \left(\frac{\lambda_j}{\lambda_1} \right)^{\nu} c_j x_{jk}} \right\}, \quad (5.12)$$

where $x_{jk} = (x_j)_k = k$ -th component of x_j . If $c_1 \neq 0$ and k is such that $x_{1k} \neq 0$, then

$$\mu_{\nu+1} = \lambda_1 + \mathcal{O} \left(\left| \frac{\lambda_2}{\lambda_1} \right|^{\nu} \right), \quad \nu = 1, 2, 3, \dots \quad (5.13)$$

Thus, the sequence (5.11) converges to the dominant eigenvalue with the same rate of convergence as for the eigenvector. Note that if $\|\cdot\|_{\infty}$ is used in the normalization of σ_{ν} , then $|\mu_{\nu+1}| = \sigma_{\nu+1}$ for ν sufficiently large. (Recall that $\sigma_{\nu+1} = \|Aq^{(\nu)}\|_{\infty}$ and $\|q^{(\nu)}\|_{\infty} = \|q^{(\nu+1)}\|_{\infty} = 1$. Also, $q_k^{(\nu+1)} = q_k^{(\nu)} = \pm 1$ in (5.11) if the maximum element is used for normalization.)

A second way of finding an approximation to λ_1 is to use the *Rayleigh quotient*, i.e., compute for any ν

$$\mu_{\nu+1} = \frac{(q^{(\nu)})^H A q^{(\nu)}}{(q^{(\nu)})^H q^{(\nu)}} = \frac{(q^{(\nu)})^H q^{(\nu+1)} \sigma_{\nu+1}}{(q^{(\nu)})^H q^{(\nu)}}. \quad (5.14)$$

Substituting (5.7) into (5.14) yields

$$\mu_{\nu+1} = \frac{\lambda_1 \sum_{j=1}^n \sum_{k=1}^n \left(\frac{\lambda_j}{\lambda_1} \right)^{\nu+1} \left(\frac{\bar{\lambda}_k}{\lambda_1} \right)^{\nu} \bar{c}_k c_j (x_k^H x_j)}{\sum_{j=1}^n \sum_{k=1}^n \left(\frac{\lambda_j}{\lambda_1} \right)^{\nu} \left(\frac{\bar{\lambda}_k}{\lambda_1} \right)^{\nu} \bar{c}_k c_j (x_k^H x_j)}. \quad (5.15)$$

It can then be shown that the estimate (5.13) holds. However, in the special case of A having orthogonal eigenvectors, i.e., $x_k^H x_j = 0$ for $j \neq k$, such as for a real symmetric or Hermitian matrix, the sequence defined by (5.15) satisfies

$$\mu_{\nu+1} = \lambda_1 + \mathcal{O} \left(\left| \frac{\lambda_2}{\lambda_1} \right|^{2\nu} \right). \quad (5.16)$$

(To see this, note that

$$\mu_{\nu+1} = \lambda_1 \left[\frac{|c_1|^2 + |c_2|^2 \left| \frac{\lambda_2}{\lambda_1} \right|^{2\nu+1} + \dots}{|c_1|^2 + |c_2|^2 \left| \frac{\lambda_2}{\lambda_1} \right|^{2\nu} + \dots} \right],$$

so

$$|\mu_{\nu+1} - \lambda_1| \leq c \left| \frac{\lambda_2}{\lambda_1} \right|^{2\nu}$$

for ν sufficiently large.)

REMARK 5.7 The more general class of normal matrices $A^H A = A A^H$ has orthogonal eigenvectors. $\square$

REMARK 5.8 If the $q^{(\nu)}$'s are normalized with respect to the ℓ_2 -norm, formula (5.14) becomes

$$|\mu_{\nu+1}| = \left(q^{(\nu)} \right)^H q^{(\nu+1)} \sigma_{\nu+1} \quad (5.17)$$

since

$$\left(q^{(\nu)} \right)^H q^{(\nu)} = 1 \quad \text{and} \quad \sigma_{\nu+1} = \|A q^{(\nu)}\|_2.$$

$\square$

REMARK 5.9 Iterations are usually terminated by a criterion such as

$$\|q^{(\nu+1)} - q^{(\nu)}\| < \epsilon,$$

where ϵ is a prescribed tolerance. $\square$

REMARK 5.10 We made two assumptions above:

- (a) A is nondefective (A has a complete set of eigenvectors.);
- (b) A has a unique dominant eigenvalue.

However, if the dominant eigenvalue is unique but not simple, the power method will still converge. Suppose that λ_1 has multiplicity r and has r linearly independent eigenvectors. Then,

$$q^{(0)} = \sum_{j=1}^r c_j x_j + \sum_{j=r+1}^n c_j x_j.$$

The sequence $q^{(\nu)}$ will converge to the direction

$$\sum_{j=1}^r c_j x_j.$$

However, if the dominant eigenvalue is not unique, e.g.

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad \lambda_1 = 1, \quad \lambda_2 = \frac{1}{2} + \frac{\sqrt{3}}{2}i, \quad \lambda_3 = \frac{1}{2} - \frac{\sqrt{3}}{2}i,$$

then the power method will fail to converge. This severely limits the applicability of the power method. □

REMARK 5.11 Once a dominant eigenvalue and eigenvector have been found, a deflation technique may be applied to define a smaller matrix whose eigenvalues are the remaining eigenvalues of A . The power method is then applied to the smaller matrix. If all eigenvalues of A are simple with different magnitudes, this procedure can be used to find all the eigenvalues. (We will come back to this later.) □

Example 5.4

(Eigenvalue computation)

$$A = \begin{pmatrix} 4 & -1 & 1 \\ -1 & 3 & -2 \\ 1 & -2 & 3 \end{pmatrix}, \quad \lambda_1 = 6, \quad \lambda_2 = 3, \quad \lambda_3 = 1, \quad x_1 = (1 \quad -1 \quad 1)^T.$$

The eigenvector approximations are

ν	$(q^{(\nu)})^T$
0	(1 0 0)
1	(1 -1/4 1/4)
4	(1 -0.8333 0.8333)
5	(1 -0.9118 0.9118)
8	(1 -0.9884 0.9884)
9	(1 -0.9942 0.9942)

Also, the eigenvalue approximations are

ν	$\mu_{\nu+1}$ (using (5.11))	$\mu_{\nu+1}$ (using (5.14), the Rayleigh Quotient)
0	4	4
4	5.6666	5.9769
8	5.9768	6.0000

5.3 The Inverse Power Method

The inverse power method has a faster rate of convergence than the power method, and can be used to compute any eigenvalue, not just the dominant one. A description of this method follows.

Let A have eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$ corresponding to linearly independent eigenvectors $x_1, x_2, \dots, x_n$. (Here, the eigenvalues are not necessarily ordered.) Then, the matrix $(A - \lambda I)^{-1}$ has eigenvalues $(\lambda - \lambda_1)^{-1}, (\lambda - \lambda_2)^{-1}, \dots, (\lambda - \lambda_n)^{-1}$, corresponding to eigenvectors $x_1, x_2, \dots, x_n$. Let $q^{(0)}$ be a starting vector, with

$$q^{(0)} = \sum_{j=1}^n c_j x_j \quad (5.18)$$

(assuming that the $x_j, j = 1, 2, \dots, n$, are linearly independent.) We let

$$q^{(1)} = (\lambda I - A)^{-1} q^{(0)} = \sum_{j=1}^n \frac{c_j x_j}{\lambda - \lambda_j}.$$

Continuing in this manner, we let

$$q^{(\nu)} = [(\lambda I - A)^{-1}]^\nu q^{(0)} = (\lambda I - A)^{-1} q^{(\nu-1)} = \sum_{j=1}^n \frac{c_j x_j}{(\lambda - \lambda_j)^\nu}. \quad (5.19)$$

Suppose that λ is a close approximation to λ_1 , i.e., λ is much nearer to λ_1 than to $\lambda_2, \dots, \lambda_n$. (It is not necessary that λ_1 be dominant.) Then, if λ_1 is simple, $(\lambda - \lambda_1)^{-1}$ is much greater than any of $(\lambda - \lambda_2)^{-1}, (\lambda - \lambda_3)^{-1}, \dots, (\lambda - \lambda_n)^{-1}$. Thus, if $c_1 \neq 0$, the term $c_1 x_1 / (\lambda - \lambda_1)^\nu$ will dominate in (5.19). If

$$|(\lambda - \lambda_2)^{-1}| \geq |(\lambda - \lambda_j)^{-1}|, \quad 3 \leq j \leq n,$$

then

$$q^{(\nu)} = \frac{1}{(\lambda - \lambda_1)^\nu} \left[c_1 x_1 + \mathcal{O} \left(\left| \frac{\lambda_1 - \lambda}{\lambda_2 - \lambda} \right|^\nu \right) \right]. \quad (5.20)$$

Thus, the iterates $q^{(\nu)}$ converge in direction to x_1 . In practice, $q^{(\nu)}$ would be normalized, as was done in the power method.

The error estimate for the inverse power method analogous to (5.10) is

$$q^{(\nu)} - \frac{x_1}{\|x_1\|} = \mathcal{O} \left(\left| \frac{\lambda - \lambda_1}{\lambda - \lambda_2} \right|^\nu \right), \quad (5.21)$$

where $q^{(\nu)}$ is normalized so that $\|q^{(\nu)}\| = 1$.

REMARK 5.12 When λ is close to λ_1 , the convergence of the inverse power method can be rapid. If $\lambda_1 = 1$ and $\lambda_2 = 0.95$, then $|\lambda_2/\lambda_1|^\nu \leq 0.1$

requires $\nu > 44$ for the power method. If $\lambda = 0.99$ in the inverse power method, then

$$\left(\frac{0.99-1}{0.99-0.95}\right)^\nu = \left(\frac{1}{4}\right)^\nu \leq 0.1 \text{ for } \nu \geq 2, \text{ and } \left(\frac{1}{4}\right)^\nu \leq 10^{-26} \text{ for } \nu > 44.$$

□

REMARK 5.13 If $\lambda = 0$, the inverse power method is simply the power method for A^{-1} , and $q^{(\nu)}$ converges to an eigenvector corresponding to the dominant eigenvalue of A^{-1} (i.e., the inverse of the eigenvalue of least magnitude of A). □

REMARK 5.14 If A is real and λ_1 is complex, then the choice λ real results in $|\lambda - \lambda_1| = |\lambda - \lambda_2|$ where $\lambda_2 = \bar{\lambda}_1$. (Complex eigenvalues occur in conjugate pairs for A real.) The inverse power method will then not converge. For this reason, the inverse power method is often regarded as a method applicable to symmetric or Hermitian matrices. □

Consider now estimation of eigenvalues. Analogous to (5.11) and (5.14), we have

$$\frac{1}{\lambda - \mu_{\nu+1}} = \frac{(q^{(\nu+1)})_k}{(q^{(\nu)})_k} = \frac{((\lambda I - A)^{-1} q^{(\nu)})_k}{(q^{(\nu)})_k} \approx \frac{1}{\lambda - \lambda_1} \quad (5.22)$$

and

$$\frac{1}{\lambda - \mu_{\nu+1}} = \frac{(q^{(\nu)})^H q^{(\nu+1)}}{(q^{(\nu)})^H q^{(\nu)}} \approx \frac{1}{\lambda - \lambda_1} \text{ (Rayleigh Quotient)}. \quad (5.23)$$

Example 5.5

$$A = \begin{pmatrix} 4 & -1 & 1 \\ 1 & 1 & 1 \\ -2 & 0 & -6 \end{pmatrix}, \quad \lambda_1 = -5.76849, \quad \lambda_2 = 3.4691, \quad \lambda_3 = 1.29923.$$

The power method requires 22 iterations to find $\lambda_1 \approx -5.7685 = \mu_{22}$ and $q^{(22)} \approx (-0.1157 \ -0.1306 \ 1)^T$. With $\lambda = -5.77$, the inverse power method requires 4 iterations to find $1/(\lambda - \mu_4) \approx -672.438$, and thus, $\mu_4 = -5.7685 \approx \lambda_1$. □

REMARK 5.15 To calculate $q^{(\nu+1)}$ from $q^{(\nu)}$, generally $(\lambda I - A)^{-1}$ is not computed, but the following system is solved for $q^{(\nu+1)}$ instead:

$$(\lambda I - A)q^{(\nu+1)} = q^{(\nu)} \quad (q^{(\nu+1)} \text{ is then scaled}). \quad (5.24)$$

In general, Gaussian elimination with pivoting is used to solve (5.24). The method initially requires $\mathcal{O}(n^3)$ multiplications, but the multipliers are saved to reduce further computations. $\square$

REMARK 5.16 The selection of λ can be based on some other method, or perhaps Gerschgorin's Theorem can be used. $\square$

5.4 Deflation

Suppose that λ_1 and x_1 are determined for an $n \times n$ matrix A using the power method or inverse power method. The goal of deflation is to find a simpler $(n-1) \times (n-1)$ matrix whose eigenvalues are those of A except for λ_1 . The power method or inverse power method can then be applied to the simpler matrix to find another eigenvalue, say λ_2 .

To begin, let x be an eigenvector of A corresponding to eigenvalue λ . Let U be an $n \times (n-1)$ matrix such that (x, U) is unitary. Since $Ax = \lambda x$ and $A(x, U) = (Ax, AU)$, we have

$$(x, U)^H A(x, U) = \begin{pmatrix} x^H \\ U^H \end{pmatrix} (\lambda x, AU) = \begin{pmatrix} \lambda x^H x & x^H AU \\ \lambda U^H x & U^H AU \end{pmatrix}.$$

Now $x^H x = 1$ and x is orthogonal to the columns of U , i.e., $U^H x = 0$. Thus,

$$(x, U)^H A(x, U) = \begin{pmatrix} \lambda & h^H \\ 0 & C \end{pmatrix}, \quad (5.25)$$

where $C = U^H AU$ is $(n-1)$ by $(n-1)$ and $h^H = x^H AU$.

The matrix in (5.25) is block triangular and has for its eigenvalues λ and the eigenvalues of C . But the matrix in (5.25) is obtained from A by a similarity transformation, so it has the same eigenvalues as A . Thus, C has the same eigenvalues as A , except for λ . We are thus left with the following question: How do we determine (x, U) ? It turns out that (x, U) can be determined by means of a Householder transformation, as we saw in §3.3.8 (starting on page 132).

Let's briefly review Householder transformations. Recall that if $\|w\|_2 = 1$, the $n \times n$ matrix $W = I - 2ww^H$ is called a Householder transformation. We saw that Householder transformations have several interesting properties:

- (a) $W = W^H$ (W is Hermitian).
- (b) $W^H W = W^2 = I$ (W is unitary).

- (c) If $x \in \mathbb{R}^n$, x_1 is the first component of x , $x \neq 0$, $\sigma = \text{sign}(x_1)\|x\|_2$ where $\text{sign}(0) = +1$, and if

$$w = x + \sigma e^{(1)} \quad \text{and} \quad \theta = \frac{1}{2}\|w\|_2^2, \quad \text{then} \quad W = I - \frac{1}{\theta}ww^H$$

is a Householder transformation, and $Wx = -\sigma e^{(1)}$.

Property (c) indicates that Householder transformations have the capability of introducing zeros into vectors.

Now recall that x is an eigenvector corresponding to λ . We will construct a Householder transformation T using property (c) such that

$$Tx = -\text{sign}(x_1)\|x\|_2 e^{(1)} = -\text{sign}(x_1)e^{(1)},$$

assuming that $\|x\|_2 = 1$. Then,

$$T^2x = x = -\text{sign}(x_1)Te^{(1)}.$$

Thus, $Te^{(1)} = -\text{sign}(x_1)x$. Therefore, the first column of T is the eigenvector $-\text{sign}(x_1)x$. Hence, $T = (-\text{sign}(x_1)x, U)$, where U is unitary. We have thus found U . In summary, the deflation procedure is:

- (i) Begin with λ , x an eigenpair of A with $\|x\|_2 = 1$.
- (ii) Compute T such that $Tx = -\text{sign}(x_1)\|x\|_2 e^{(1)}$.
 $(\sigma \leftarrow \text{sign}(x_1)\|x\|_2, w \leftarrow x + \sigma e^{(1)}, \theta \leftarrow (1/2)\|w\|_2^2, T \leftarrow I - ww^H/\theta.)$
- (iii) Find U through the relation $T = (-\text{sign}(x_1)x, U)$, where U is $n \times (n-1)$.
- (iv) $C \leftarrow U^H AU$.

By (5.25),

$$T^H AT = \begin{pmatrix} \lambda & h^H \\ 0 & C \end{pmatrix},$$

and the $(n-1) \times (n-1)$ matrix C has the same eigenvalues as A except for λ_1 . The matrix C can then be used in conjunction with the power method or inverse power method to find a second eigenvalue of A .

REMARK 5.17 Other deflation methods, such as Wielandt's deflation method, are available, but it can be shown that the deflation method we have just described is not seriously affected by rounding errors. $\square$

5.5 The QR Method

The QR method is an iterative method for reducing a matrix to triangular form using orthogonal similarity transformations. When applied to a Hessenberg matrix,³ the QR method is very competitive. The QR method is one of the best known methods for finding all the eigenvalues of a matrix. The eigenvectors may be found by transforming back the eigenvectors of the final triangular matrix.

Computing eigenvalues and eigenvectors by the QR method involves two steps:

1. reducing the matrix to Hessenberg form, and
2. iteration by QR decompositions.

5.5.1 Reduction to Hessenberg or Tridiagonal Form

The number of operations required to complete one QR transformation is proportional to n^3 for a full matrix but only to n^2 for a Hessenberg matrix. Thus, in the QR method, the matrix A is first transformed to (upper) Hessenberg form, then the QR method is applied to the upper Hessenberg matrix.

DEFINITION 5.3 *A matrix A is (upper) Hessenberg if A has the form*

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & & a_{1n} \\ a_{21} & a_{22} & \cdots & & a_{2n} \\ 0 & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & a_{nn-1} & a_{nn} \end{pmatrix},$$

that is, if A has nonzeros only on and above the main diagonal and in the positions one index below the main diagonal.

To reduce A to Hessenberg form, we employ Householder transformations in the following procedure. Let $A_1 = A$ be a given $n \times n$ matrix. At the k -th step, the reduced matrix A_k has the form

$$A_k = \begin{pmatrix} A_{11}^{(k)} & a_{12}^{(k)} & A_{13}^{(k)} \\ 0 & a_{22}^{(k)} & A_{23}^{(k)} \end{pmatrix}, \quad (5.26)$$

³We will define “Hessenberg matrix” shortly.

where $A_{11}^{(k)}$ is $k \times (k-1)$, $a_{12}^{(k)}$ is $k \times 1$, $a_{22}^{(k)}$ is $(n-k) \times 1$, $A_{13}^{(k)}$ is $k \times (n-k)$ and $A_{23}^{(k)}$ is $(n-k) \times (n-k)$. Let H_k be an $(n-k) \times (n-k)$ Householder matrix such that

$$H_k a_{22}^{(k)} = \pm \|a_{22}^{(k)}\| e^{(1)}, \text{ where } e^{(1)} \text{ is the first unit vector in } \mathbb{R}^{n-k}, \quad (5.27)$$

and let

$$U_k = \begin{pmatrix} I_k & 0 \\ 0 & H_k \end{pmatrix}. \quad (\text{Recall that } H_k^H = H_k \text{ and } H_k^2 = I.) \quad (5.28)$$

(See Lemma 3.3 on page 134 for details on construction of the transformation.) Then,

$$A_{k+1} = U_k^H A U_k = U_k A U_k = \begin{pmatrix} A_{11}^{(k)} & a_{12}^{(k)} & A_{13}^{(k)} H_k \\ 0 & \pm \|a_{22}^{(k)}\| e_1^{(1)} & H_k A_{23}^{(k)} H_k \end{pmatrix}, \quad (5.29)$$

which carries the reduction to Hessenberg form one step further. The complete reduction requires approximately $\frac{5}{3}n^3$ multiplications (Exercise 12 on page 321). The final Hessenberg matrix is the matrix A_{n-1} given by

$$A_{n-1} = U_{n-2}^H U_{n-3}^H \cdots U_1^H A U_1 U_2 \cdots U_{n-2}. \quad (5.30)$$

Of course, if $U = U_1 U_2 \cdots U_{n-2}$, then U is orthogonal, and A_{n-1} is similar to A , i.e.,

$$A_{n-1} = U^H A U = U^{-1} A U = U A U. \quad (5.31)$$

REMARK 5.18 If A is Hermitian, then (5.31) implies that A_{n-1} is Hermitian. Since a Hermitian (upper) Hessenberg matrix is tridiagonal, we see that A_{n-1} is tridiagonal. Furthermore, for A Hermitian, the algorithm to find A_{n-1} can be implemented in such a way that only about $\frac{2}{3}n^3$ multiplications are required (Exercise 13 on page 321). $\square$

5.5.2 The QR Method with Origin Shifts

We first describe the algorithm. Let $A = A_0$ be a given $n \times n$ complex matrix, preferably in Hessenberg or tridiagonal form. Then, the QR algorithm produces a sequence of matrices $A_0, A_1, A_2, \dots$ determined as follows:

- Given A_ν , determine the scalar *origin shift* μ_ν from the elements of A_ν ,
- Factor the matrix $A_\nu - \mu_\nu I$ into the form

$$A_\nu - \mu_\nu I = Q_\nu R_\nu, \quad (5.32)$$

where Q_ν is unitary and R_ν is upper triangular. (This is the orthogonal triangularization or QR decomposition of $A_\nu - \mu_\nu I$.)

(c) Compute $A_{\nu+1}$ by the formula

$$A_{\nu+1} = R_\nu Q_\nu + \mu_\nu I. \quad (5.33)$$

REMARK 5.19 As the iteration continues, the scalars μ_ν converge to an eigenvalue of A . (We will explain how to choose μ_ν later.) $\square$

REMARK 5.20 By Theorem 3.13 (page 136), the factorization (5.32) exists. In addition, if $A_\nu - \mu_\nu I$ is nonsingular, the factorization is unique. To see this, let $A_\nu - \mu_\nu I = \hat{Q}_\nu \hat{R}_\nu$ with $\hat{Q}_\nu$ and $\hat{R}_\nu$ having the required properties. Then,

$$(A_\nu - \mu_\nu I)^H (A_\nu - \mu_\nu I) = B_\nu = \hat{R}_\nu^H \hat{Q}_\nu^H \hat{Q}_\nu \hat{R}_\nu = \hat{R}_\nu^H \hat{R}_\nu,$$

and by (5.32), $B_\nu = R_\nu^H R_\nu$. Thus, B_ν has the Cholesky factorizations $B_\nu = \hat{R}_\nu^H \hat{R}_\nu = R_\nu^H R_\nu$. By uniqueness of the Cholesky factorization, $R_\nu = \hat{R}_\nu$; $\hat{Q}_\nu = Q_\nu$ then follows from nonsingularity of R_ν . $\square$

REMARK 5.21 From (5.32) and (5.33), we see that $R_\nu = Q_\nu^H (A_\nu - \mu_\nu I)$ and $A_{\nu+1} = Q_\nu^H (A_\nu - \mu_\nu I) Q_\nu + \mu_\nu I = Q_\nu^H A_\nu Q_\nu$. Thus, $A_{\nu+1}$ is unitarily similar to A_ν and therefore has the same eigenvalues. Also, if A_ν is (upper) Hessenberg, then $A_{\nu+1}$ is (upper) Hessenberg (Exercise 14 on page 321). $\square$

REMARK 5.22 A single iteration of the QR method requires a multiple of n^3 multiplications. However, if A is (upper) Hessenberg, the method only requires a multiple of n^2 multiplications, and if A is tridiagonal, only $\mathcal{O}(n)$ multiplications are required (Exercise 15 on page 321). $\square$

5.5.2.1 Convergence of the QR Method

We have the following question: Why does the QR method converge? That is, why does the sequence $\{A_\nu\}$ tend to a triangular matrix that is unitarily similar to A_0 ?

We will see that the QR method is related to the power method and also the inverse power method. Let

$$\tilde{Q}_\nu = Q_0 Q_1 \dots Q_\nu \quad \text{and} \quad \tilde{R}_\nu = R_\nu R_{\nu-1} \dots R_0. \quad (5.34)$$

Since $A_{\nu+1} = Q_\nu^H A_\nu Q_\nu$, we have

$$A_{\nu+1} = \tilde{Q}_\nu^H A \tilde{Q}_\nu \quad (5.35)$$

and, since $\tilde{Q}_\nu$ is unitary,

$$(A_{\nu+1} - \mu_\nu I) = \tilde{Q}_\nu^H (A - \mu_\nu I) \tilde{Q}_\nu. \quad (5.36)$$

The following result connects the power method to the QR method.

PROPOSITION 5.5

In the above notation

$$\tilde{Q}_\nu \tilde{R}_\nu = (A - \mu_\nu I)(A - \mu_{\nu-1} I) \dots (A - \mu_0 I). \quad (5.37)$$

PROOF For $\nu = 0$, (5.37) simply defines Q_0 and R_0 . Assume now (5.37) holds for $\nu = k - 1$. From (5.33) and (5.36), we have

$$\begin{aligned} R_k &= (A_{k+1} - \mu_k I) Q_k^H \\ &= \tilde{Q}_k^H (A - \mu_k I) \tilde{Q}_k Q_k^H \\ &= \tilde{Q}_k^H (A - \mu_k I) \tilde{Q}_{k-1}. \end{aligned}$$

Multiplying both sides on the right by $\tilde{R}_{k-1}$ results in

$$\tilde{R}_k = \tilde{Q}_k^H (A - \mu_k I) \tilde{Q}_{k-1} \tilde{R}_{k-1},$$

so

$$\tilde{Q}_k \tilde{R}_k = (A - \mu_k I) \tilde{Q}_{k-1} \tilde{R}_{k-1}.$$

Therefore, (5.37) holds by induction. $\square$

To see the significance of (5.37), consider the unshifted QR method, i.e., set $0 = \mu_0 = \mu_1 = \dots = \mu_\nu$. Then, (5.37) becomes $A^{\nu+1} = \tilde{Q}_\nu \tilde{R}_\nu$, i.e., $\tilde{Q}_\nu \tilde{R}_\nu$ is the QR decomposition of $A^{\nu+1}$. Also, since $\tilde{R}_\nu e^{(1)} = \tilde{r}_{11}^{(\nu)} e^{(1)}$ (because $\tilde{R}_\nu$ is upper triangular), the first column of $\tilde{Q}_\nu$ satisfies

$$\tilde{Q}_\nu \tilde{R}_\nu e^{(1)} = \tilde{r}_{11}^{(\nu)} q_\nu^{(1)} = A^{\nu+1} e^{(1)}.$$

Thus, $q_\nu^{(1)}$ is the vector obtained by applying $(\nu+1)$ steps of the power method to the initial vector $e^{(1)}$. If A has a dominant eigenvalue λ_1 and $e^{(1)}$ has a nonzero component in the direction of the eigenvector corresponding to λ_1 , then $q_\nu^{(1)}$ approaches that eigenvector.

Now partition A_ν into the form

$$A_\nu = \begin{pmatrix} a_{11}^{(\nu)} & h_\nu^H \\ g_\nu & C_\nu \end{pmatrix},$$

and do likewise for $A_{\nu+1}$. Analogous to the unitary matrix in deflation (see page 305), we partition Q_ν into its first column $q_\nu^{(1)}$ and remaining columns U_ν to obtain

$$\begin{aligned} A_{\nu+1} &= Q_\nu^H A_\nu Q_\nu = (q_\nu^{(1)}, U_\nu)^H A_\nu (q_\nu^{(1)}, U_\nu) \\ &= \begin{pmatrix} q_\nu^{(1)H} A_\nu q_\nu^{(1)} & q_\nu^{(1)H} A_\nu U_\nu \\ U_\nu^H A_\nu q_\nu^{(1)} & U_\nu^H A_\nu U_\nu \end{pmatrix} = \begin{pmatrix} a_{11}^{(\nu+1)} & h_{\nu+1}^H \\ g_{\nu+1} & C_{\nu+1} \end{pmatrix}. \end{aligned}$$

But $A_\nu q_\nu^{(1)} \rightarrow \lambda_1 q^{(1)}$ and $U_\nu^H \lambda_1 q_\nu^{(1)} = \lambda_1 U_\nu^H q_\nu^{(1)} = 0$, since Q_ν^H is unitary. Thus, $U_\nu^H A_\nu q_\nu^{(1)} = g_{\nu+1} \rightarrow 0$ as $\nu \rightarrow \infty$ and

$$a_{11}^{(\nu+1)} = (q_\nu^{(1)})^H A_\nu q_\nu^{(1)} \rightarrow \lambda_1 (q_\nu^{(1)})^H q_\nu^{(1)} \rightarrow \lambda_1 \quad \text{as } \nu \rightarrow \infty.$$

Now let's consider the shifted QR method in relation to the inverse power method. From (5.32), we have

$$Q_\nu = (A_\nu^H - \bar{\mu}_\nu I)^{-1} R_\nu^H,$$

since $Q_\nu^H = R_\nu (A_\nu - \mu_\nu I)^{-1}$. Since R_ν^H is lower triangular, $R_\nu^H e^{(n)} = \bar{r}_{nn} e^{(n)}$, where r_{ij} is the i, j -th element of R_ν . Now, let $q_\nu^{(i)}$ represent the i -th column of Q_ν . Then,

$$q_\nu^{(n)} = Q_\nu e^{(n)} = (A_\nu^H - \bar{\mu}_\nu I)^{-1} R_\nu^H e^{(n)} = \bar{r}_{nn} (A_\nu^H - \bar{\mu}_\nu I)^{-1} e^{(n)}. \quad (5.38)$$

Thus, the last column of Q_ν , that is, $q_\nu^{(n)}$, is the approximate eigenvector of A_ν^H that results from applying one step of the inverse power method with shift $\bar{\mu}_\nu$ to the vector $e^{(n)}$. This observation has the following consequences in terms of $A_{\nu+1}$. Partition A_ν into the form

$$A_\nu = \begin{pmatrix} C_\nu & h_\nu \\ g_\nu^H & a_{nn}^\nu \end{pmatrix}, \quad (5.39)$$

with $A_{\nu+1}$ partitioned accordingly, i.e.,

$$A_{\nu+1} = \begin{pmatrix} C_{\nu+1} & h_{\nu+1} \\ g_{\nu+1}^H & a_{nn}^{\nu+1} \end{pmatrix}.$$

Consider $A_{\nu+1} = Q_\nu^H A_\nu Q_\nu$ with

$$Q_\nu = (U_\nu, q_\nu^{(n)}).$$

Then,

$$A_{\nu+1} = (U_\nu, q_\nu^{(n)})^H A_\nu (U_\nu, q_\nu^{(n)}) = \begin{pmatrix} U_\nu^H A_\nu U_\nu & U_\nu^H A_\nu q_\nu^{(n)} \\ q_\nu^{(n)H} A_\nu U_\nu & q_\nu^{(n)H} A_\nu q_\nu^{(n)} \end{pmatrix}.$$

If $\bar{\mu}_\nu$ is near an eigenvalue of A^H , then $q_\nu^{(n)}$ will be nearer an accurate eigenvector than $e^{(n)}$; thus, $g_{\nu+1}^H = q_\nu^{(n)H} A_\nu U_\nu$ gives

$$g_{\nu+1} = U_\nu^H A_\nu^H q_\nu^{(n)} \approx \lambda_n U_\nu^H q_\nu^{(n)} = 0,$$

since the columns of U_ν are orthogonal to $q_\nu^{(n)}$. Thus, $\|g_{\nu+1}\|$ will be smaller than $\|g_\nu\|$. In fact, judging from the convergence rate of the inverse power

method, the vectors g_ν may approach zero rapidly if the μ_ν 's ever approximate an eigenvalue.

In (5.38), the natural choice for $\bar{\mu}_\nu$ is the Rayleigh quotient $(e^{(n)})^H A_\nu^H e^{(n)}$, i.e.,

$$\mu_\nu = a_{nn}^{(\nu)}. \quad (5.40)$$

Notice that, then,

$$a_{nn}^{(\nu+1)} = (q_\nu^{(n)})^H A_\nu q_\nu^{(n)} \approx \lambda_n.$$

Once g_ν is “sufficiently small,” we may neglect it and restart the QR process with the smaller matrix C_ν . Thus, the QR method “naturally deflates” A .

REMARK 5.23 Suppose that A is real and (upper) Hessenberg and we wish to work in real arithmetic. Then, we must choose real shifts μ_ν ; therefore, we cannot approximate a complex eigenvalue well. In the next section, we present a variant of the QR method, in which complex eigenvalues (which occur in conjugate pairs since A is real), can be determined by solving 2×2 eigenvalue problems. $\square$

REMARK 5.24 If A is symmetric, then its eigenvalues are real, and we do not have the problem of Remark 5.23. Therefore, the method of this section, coupled with preliminary reduction to tridiagonal form, is an effective and popular way to compute all the eigenvalues of a symmetric matrix. $\square$

REMARK 5.25 In the symmetric case, if we choose the shifts μ_ν to be $a_{nn}^{(\nu)}$, then the process usually converges rapidly. In practice, the shift is chosen to be the eigenvalue of

$$\begin{pmatrix} a_{n-1,n-1}^{(\nu)} & a_{n-1,n}^{(\nu)} \\ a_{n,n-1}^{(\nu)} & a_{n,n}^{(\nu)} \end{pmatrix}$$

closest to $a_{nn}^{(\nu)}$. Then it can be shown that the QR method always converges and generally converges rapidly. $\square$

REMARK 5.26 Because of the relationship of the QR method to the power and inverse power methods, the QR method is substantially slowed down if the eigenvalues of matrix A are packed closely together. That is, the QR method is most effective when the eigenvalues of A are separated. $\square$

5.5.3 The Double QR Algorithm

Since this algorithm uses only real arithmetic, this algorithm is advantageous when A is real and not all the eigenvalues of A are real. Consider two

steps of the QR method, i.e., (5.32) and (5.33); in other words,

$$\begin{cases} A_\nu - \mu_\nu I = Q_\nu R_\nu \\ A_{\nu+1} = R_\nu Q_\nu + \mu_\nu I \\ A_{\nu+1} - \mu_{\nu+1} I = Q_{\nu+1} R_{\nu+1} \\ A_{\nu+2} = R_{\nu+1} Q_{\nu+1} + \mu_{\nu+1} I. \end{cases} \quad (5.41)$$

From (5.41), we have $A_{\nu+2} = Q_{\nu+1}^H A_{\nu+1} Q_{\nu+1} = Q_{\nu+1}^H Q_\nu^H A_\nu Q_\nu Q_{\nu+1}$. Thus,

$$A_{\nu+2} = (Q_\nu Q_{\nu+1})^H A_\nu (Q_\nu Q_{\nu+1}). \quad (5.42)$$

An interesting result is the fact that

$$\begin{aligned} Q_\nu Q_{\nu+1} R_{\nu+1} R_\nu &= Q_\nu (A_{\nu+1} - \mu_{\nu+1} I) R_\nu \\ &= Q_\nu A_{\nu+1} R_\nu - \mu_{\nu+1} Q_\nu R_\nu \\ &= Q_\nu (R_\nu Q_\nu + \mu_\nu I) R_\nu - \mu_{\nu+1} Q_\nu R_\nu \\ &= Q_\nu R_\nu Q_\nu R_\nu + \mu_\nu Q_\nu R_\nu - \mu_{\nu+1} Q_\nu R_\nu \\ &= (Q_\nu R_\nu + \mu_\nu I) Q_\nu R_\nu - \mu_{\nu+1} Q_\nu R_\nu \\ &= A_\nu Q_\nu R_\nu - \mu_{\nu+1} Q_\nu R_\nu \\ &= (A_\nu - \mu_{\nu+1} I) Q_\nu R_\nu \\ &= (A_\nu - \mu_{\nu+1} I) (A_\nu - \mu_\nu I) \end{aligned}$$

Thus,

$$Q_\nu Q_{\nu+1} R_{\nu+1} R_\nu = (A_\nu - \mu_{\nu+1} I) (A_\nu - \mu_\nu I). \quad (5.43)$$

Thus, if A is real and $\mu_{\nu+1} = \bar{\mu}_\nu$, the matrix on the right in (5.43) is real, and this implies that $Q_\nu Q_{\nu+1}$ and $R_{\nu+1} R_\nu$ are real. (If A is real, the Householder transformations used to construct the QR decomposition are real.) Hence, each element of the sequence $A_1, A_3, A_5, \dots$ is real, since A_1 is real by assumption and $A_3 = (Q_1 Q_2)^H A_1 (Q_1 Q_2)$, $A_5 = (Q_3 Q_4)^H A_3 (Q_3 Q_4)$, $\dots$ by (5.42).

We can avoid computing $A_2, A_4, \dots$ with the following procedure. We need $Q_\nu Q_{\nu+1}$ to compute $A_{\nu+2}$ by (5.42), i.e., $A_{\nu+2} = (Q_\nu Q_{\nu+1})^H A_\nu (Q_\nu Q_{\nu+1})$. However, we can find $Q_\nu Q_{\nu+1}$ without finding $A_{\nu+1}$. We decompose $(A_\nu - \mu_\nu I)(A_\nu - \bar{\mu}_\nu I) = \tilde{Q}_\nu \tilde{R}_\nu$. Then, $\tilde{Q}_\nu = Q_\nu Q_{\nu+1}$ by (5.43), $A_{\nu+2} = \tilde{Q}_\nu^H A_\nu \tilde{Q}_\nu$ by (5.42), and we have found $A_{\nu+2}$ with real arithmetic and without computing $A_{\nu+1}$. Thus, $A_1, A_3, A_5, \dots$ can be computed using this procedure. The price we pay for this convenience is at step ν of the double QR algorithm, $(A_\nu - \mu_\nu I)(A_\nu - \bar{\mu}_\nu I)$ must be computed.

REMARK 5.27 The shifts μ_ν and $\mu_{\nu+1}$ are usually chosen to be the eigenvalues of

$$\begin{pmatrix} a_{n-1,n-1}^{(\nu)} & a_{n-1,n}^{(\nu)} \\ a_{n,n-1}^{(\nu)} & a_{nn}^{(\nu)} \end{pmatrix}. \quad (5.44)$$

It is easy to verify that μ_ν and $\mu_{\nu+1}$ satisfy

$$\mu_\nu + \mu_{\nu+1} = a_{n-1,n-1}^{(\nu)} + a_{nn}^{(\nu)}$$

and

$$\mu_\nu \mu_{\nu+1} = a_{nn}^{(\nu)} a_{n-1,n-1}^{(\nu)} - a_{n-1,n}^{(\nu)} a_{n,n-1}^{(\nu)},$$

and also

$$\begin{aligned} (A_\nu - \mu_\nu I)(A_\nu - \mu_{\nu+1} I) &= A_\nu^2 - (a_{n-1,n-1}^{(\nu)} + a_{nn}^{(\nu)}) A_\nu \\ &\quad + (a_{nn}^{(\nu)} a_{n-1,n-1}^{(\nu)} - a_{n-1,n}^{(\nu)} a_{n,n-1}^{(\nu)}) I. \end{aligned}$$

Hence, if the eigenvalues of (5.44) are complex conjugates, i.e., $\bar{\mu}_\nu = \mu_{\nu+1}$, then the double QR method accomplishes in real arithmetic the effect of performing single shifts, with one shift μ_ν and one shift $\bar{\mu}_\nu$. $\square$

REMARK 5.28 Of course, since the elements of A_ν are real for ν odd, the element $a_{nn}^{(\nu)}$ cannot converge to a complex eigenvalue. What happens is that the 2×2 submatrix in the trailing position eventually converges. The eigenvalues of the 2×2 matrix approximate a pair of complex conjugate eigenvalues of the original matrix. Thus, the double QR method will eventually produce a matrix orthogonally similar to the original matrix, and the eigenvalues can be found by solving 2×2 eigenvalue problems for complex eigenvalues or 1×1 problems for real eigenvalues. $\square$

REMARK 5.29 The double QR method is not guaranteed to converge. For instance, the matrix

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

is left unchanged by a double QR step, and therefore none of the subdiagonal elements converge to zero. In practice, when the method does not converge, a couple of applications of the single shift QR method with randomly chosen shifts generally produces a matrix for which the double QR method will converge. $\square$

Example 5.6

$$A = \frac{1}{(25)^2} \begin{pmatrix} 1490 & 695 & -600 \\ 695 & 1635 & -175 \\ -600 & -175 & 2500 \end{pmatrix}, \quad \begin{cases} \lambda_1 = 3 + \sqrt{3} = 4.7321, \\ \lambda_2 = 3.0, \\ \lambda_3 = 3 - \sqrt{3} = 1.2679. \end{cases}$$

The Householder transformation

$$U = \frac{1}{25} \begin{pmatrix} 7 & -24 & 0 \\ -24 & -7 & 0 \\ 0 & 0 & 25 \end{pmatrix}$$

transforms A to Hessenberg form

$$\tilde{A} = U^H A U = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & 1 \\ 0 & 1 & 4 \end{pmatrix}.$$

Choosing the shifts $\mu_\nu = a_{nn}^\nu$ (see Eq. (5.40), page 312) gives the following iterates in the QR method:

$$\begin{aligned} \tilde{A}_1 &= \begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & 1 \\ 0 & 1 & 4 \end{pmatrix}, \quad \tilde{A}_2 = \begin{pmatrix} 1.4000 & 0.4899 & 0 \\ 0.4899 & 3.2667 & 0.7454 \\ 0 & 0.7454 & 4.3333 \end{pmatrix}, \\ \tilde{A}_3 &= \begin{pmatrix} 1.2915 & 0.2017 & 0 \\ 0.2017 & 3.0202 & 0.2724 \\ 0 & 0.2724 & 4.6884 \end{pmatrix}, \quad \tilde{A}_4 = \begin{pmatrix} 1.2739 & 0.0993 & 0 \\ 0.0993 & 2.9943 & 0.0072 \\ 0 & 0.0072 & 4.7320 \end{pmatrix}, \\ \tilde{A}_5 &= \begin{pmatrix} 1.2694 & 0.0498 & 0 \\ 0.0498 & 2.9986 & 0 \\ 0 & 0 & 4.7321 \end{pmatrix}. \end{aligned}$$

At this point, the QR method can be applied to the smaller 2×2 matrix

$$\begin{pmatrix} 1.2694 & 0.0498 \\ 0.0498 & 2.9986 \end{pmatrix}.$$

□

5.6 Jacobi Diagonalization (Jacobi Method)

The Jacobi method for computing eigenvalues is one of the oldest numerical methods for the eigenvalue problem. It was replaced by the QR algorithm as the method of choice in the 1960's. However, it is making a comeback due to its adaptability to parallel computers [40, 97]. We give a brief description of the Jacobi method in this section. Let $A^{(1)} = A$ be an $n \times n$ symmetric matrix. The procedure consists of

$$A^{(k+1)} = N_k^H A^{(k)} N_k \quad (5.45)$$

where the N_k are unitary matrices that eliminate the off-diagonal element of largest modulus.

REMARK 5.30 It can be shown that if $a_{pq}^{(k)}$ is the off-diagonal element of largest modulus, one transformation increases the sum of the squares of the diagonal elements by $2a_{pq}^2$ and at the same time decreases the sum of the squares of the off-diagonal elements by the same amount. Thus, $A^{(k)}$ tends to a diagonal matrix as $k \rightarrow \infty$. $\square$

REMARK 5.31 Since $A^{(1)}$ is symmetric, $A^{(2)} = N_1^H A^{(1)} N_1$ is symmetric, so $A^{(3)}, A^{(4)} \dots$ are symmetric. Also, since $A^{(k+1)}$ is similar to $A^{(k)}$, $A^{(k+1)}$ has the same eigenvalues as $A^{(k)}$, and hence has the same eigenvalues as A . $\square$

We now consider how to find N_k such that the largest off-diagonal element of $A^{(k)}$ is eliminated. Let

$$A^{(k)} = \begin{pmatrix} a_{11}^{(k)} & \dots & a_{1n}^{(k)} \\ & \ddots & \\ a_{n1}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix},$$

and suppose that $|a_{pq}^{(k)}| \geq |a_{ij}^{(k)}|$ for $1 \leq i, j \leq n$. Let

$$N_k = \left(\begin{array}{c|c|c|c|c} 1 & \begin{array}{c} p\text{-th} \\ \text{col.} \end{array} & & \begin{array}{c} q\text{-th} \\ \text{col.} \end{array} & \\ & \downarrow & & \downarrow & \\ & & & & \\ \hline \begin{array}{c} p\text{-th} \\ \text{row} \end{array} \rightarrow & \cos(\alpha_k) & & -\sin(\alpha_k) & \\ & & 1 & & \\ \hline & & \ddots & & \\ & & & 1 & \\ \hline \begin{array}{c} q\text{-th} \\ \text{row} \end{array} \rightarrow & \sin(\alpha_k) & & \cos(\alpha_k) & \\ & & & & 1 \\ & & & & \ddots \\ & & & & 1 \end{array} \right).$$

N_k is a Givens transformation (also called a plane rotator or Jacobi rotator), as we explained starting on page 137. Note that $N_k^H N_k = I$ (N_k is unitary). When $A^{(k+1)}$ is constructed, only rows p and q and columns p and q of $A^{(k+1)}$ are different from those of $A^{(k)}$. The choice for α_k is such that $a_{pq}^{(k+1)} = 0$. That is, since

$$a_{pq}^{(k+1)} = (-a_{pp}^{(k)} + a_{qq}^{(k)}) \cos \alpha_k \sin \alpha_k + a_{pq}^{(k)} (\cos^2 \alpha_k - \sin^2 \alpha_k) = 0,$$

$\cos \alpha_k$ and $\sin \alpha_k$ are chosen so that

$$\cos^2 \alpha_k = \frac{1}{2} + \frac{a_{pp}^{(k)} - a_{qq}^{(k)}}{2r}, \quad \sin^2 \alpha_k = \frac{1}{2} - \frac{a_{pp}^{(k)} - a_{qq}^{(k)}}{2r}, \quad \text{and}$$

$$\sin \alpha_k \cos \alpha_k = \frac{a_{pq}^{(k)}}{r},$$

where $r^2 = (a_{pp}^{(k)} - a_{qq}^{(k)})^2 + 4(a_{pq}^{(k)})^2$.

In summary, the Jacobi computational algorithm consists of the following steps, where the third step provides stability with respect to rounding errors:

(1) At step k , find $a_{pq}^{(k)}$ such that $p \neq q$ and $|a_{pq}^{(k)}| \geq |a_{ij}^{(k)}|$ for $1 \leq i, j \leq n$, $i \neq j$.

(2) Set $r = \sqrt{(a_{pp}^{(k)} - a_{qq}^{(k)})^2 + 4(a_{pq}^{(k)})^2}$ and $t = 0.5 + (a_{pp}^{(k)} - a_{qq}^{(k)})/2r$.

(3) Set $\text{chk} = (a_{pp}^{(k)} - a_{qq}^{(k)})$.
IF $\text{chk} \geq 0$ *THEN*

set $c = \sqrt{t}$ and $s = a_{pq}^{(k)}/(rc)$,

ELSE

set $s = \sqrt{1-t}$ and $c = a_{pq}^{(k)}/(rs)$.

END IF

(4) Set

$$N_{i,j} = \begin{cases} 1 & \text{if } i = j, \\ c & \text{if } i = p, j = p, \\ -s & \text{if } i = p, j = q, \\ s & \text{if } i = q, j = p, \\ c & \text{if } i = q, j = q, \\ 0 & \text{otherwise.} \end{cases}$$

(5) Set $A^{(k+1)} = N^T A^{(k)} N$.

(6) Go to Step (1) until $|a_{pq}^{(k)}| < \varepsilon$.

Example 5.7

$$A = A^{(1)} = \begin{pmatrix} 5 & 1 & 0 \\ 1 & 5 & 2 \\ 0 & 2 & 5 \end{pmatrix}, \quad N_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}.$$

Then,

$$A^{(2)} = N_1^H A^{(1)} N_1 = \begin{pmatrix} 5 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 3 & 0 \\ \frac{1}{\sqrt{2}} & 0 & 7 \end{pmatrix}.$$

Notice that the sum of the squares of the diagonal elements of $A^{(2)}$ is 8 more than the sum of the squares of the diagonal elements of $A^{(1)}$, and the sum of the squares of the off-diagonal elements of $A^{(2)}$ is 8 less than the sum of the squares of the off-diagonal elements of $A^{(1)}$. $\square$

5.7 Simultaneous Iteration (Subspace Iteration)

Simultaneous iteration methods are extensions of the power method in which several vectors are processed simultaneously in such a way that the subspace spanned by these vectors converges to the subspace of the dominant set of eigenvectors of the matrix.

In this section, it is assumed that the $n \times n$ matrix A has linearly independent eigenvectors $v_1, v_2, \dots, v_n$ and associated eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$ satisfying

$$|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_k| > |\lambda_{k+1}| \geq \dots \geq |\lambda_n| > 0.$$

Let $\gamma = \text{span}(v_1, v_2, \dots, v_k)$; γ is a k -dimensional subspace of $\mathbb{C}^n$.

In simultaneous iteration methods, a matrix $U^0 = (u_1^{(0)}, u_2^{(0)}, \dots, u_k^{(0)})$ is randomly selected. At the m -th step, $V^{(m)} = AU^{(m)}$. To prevent all the trial vectors from converging to the dominant eigenvector, a new set of trial vectors $U^{(m+1)}$ is obtained from $V^{(m)}$ by an orthonormalization procedure. The following procedure may be used:

- (a) Let $u_1^{(0)}, u_2^{(0)}, \dots, u_k^{(0)}$ be a set of orthonormal vectors for the subspace

$$\gamma_0 = \text{span}(u_1^{(0)}, u_2^{(0)}, \dots, u_k^{(0)}).$$

- (b) For $m = 0, 1, 2, \dots$

- (i) Calculate $Au_1^{(m)}, Au_2^{(m)}, \dots, Au_k^{(m)}$, which is a basis for

$$\gamma_{m+1} = \text{span} \left(Au_1^{(m)}, Au_2^{(m)}, \dots, Au_k^{(m)} \right).$$

- (ii) Orthonormalize the above vectors by the Gram–Schmidt process to obtain

$$u_1^{(m+1)}, u_2^{(m+1)}, \dots, u_k^{(m+1)}$$

and

$$\gamma_{m+1} = \text{span} \left(u_1^{(m+1)}, u_2^{(m+1)}, \dots, u_k^{(m+1)} \right).$$

(Notice that the Gram–Schmidt procedure preserves the subspace.)

It is straightforward to see that $u_\ell^{(m)} \rightarrow u_\ell$ as $m \rightarrow \infty$, where

$$\text{span}(u_1, u_2, \dots, u_k) = \gamma = \text{span}(v_1, v_2, \dots, v_k).$$

(See [97].) To see this, let

$$\gamma_0 = \text{span} \left(\sum_{i=1}^n c_{i1} v_i, \sum_{i=1}^n c_{i2} v_i, \dots, \sum_{i=1}^n c_{ik} v_i \right).$$

Then,

$$\begin{aligned} \gamma_m &= \text{span} \left(\sum_{i=1}^n c_{i1} \lambda_i^m v_i, \sum_{i=1}^n c_{i2} \lambda_i^m v_i, \dots, \sum_{i=1}^n c_{ik} \lambda_i^m v_i \right) \\ &\rightarrow \gamma = \text{span}(v_1, v_2, \dots, v_k), \quad \text{as } m \rightarrow \infty, \end{aligned}$$

since $|\lambda_k| > |\lambda_{k+1}|$. (Notice also, if A is symmetric, then the first k dominant eigenvectors of A are u_ℓ , $\ell = 1, 2, \dots, k$.)

5.8 Exercises

1. If

$$A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix},$$

then

- Use the Gerschgorin theorem to bound the eigenvalues of A .
- Compute the eigenvalues and eigenvectors of A directly from the definition, and compare with the results you obtained by using Gerschgorin's circle theorem.

2. Consider

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

- Compute the eigenvalues of A directly from the definition.
- Apply the Gerschgorin circle theorem to the matrix A .

- (c) Why is A not a counterexample to the Gerschgorin theorem for Hermitian matrices (Corollary 5.1)?

3. Let

$$A = \begin{pmatrix} 0 & \frac{1}{4} & \frac{1}{5} \\ -\frac{1}{4} & 0 & \frac{1}{3} \\ -\frac{1}{5} & -\frac{1}{3} & 0 \end{pmatrix}.$$

Show that the spectral radius $\rho(A) < 1$.

4. Let A be a diagonally dominant matrix. Can zero be in the spectrum of A ?
5. Prove Theorem 3.21 (on page 160), using notation and procedures from this chapter.
Hint: You may wish to try proving the theorem first in the case that there are n distinct eigenvalues. You may also wish to consider the Schur decomposition (on page 101), although this is not the only way this theorem can be proven.
6. Apply several iterations of the power method to the matrix from Problem 1 on page 319, using several different starting vectors. Compare the results to the results you obtained from Problem 1 on page 319.
7. Let A be a real symmetric $n \times n$ matrix with dominant eigenvalues $\lambda_1 = 1$ and $\lambda_2 = -1$. Show that the power method fails to converge. Determine the behavior of successive iterates of the power method.
8. Suppose that λ_1 and x_1 have been obtained for a real symmetric $n \times n$ matrix A using the power method, and

$$|\lambda_1| > |\lambda_2| > |\lambda_3| \geq |\lambda_4| \geq \dots \geq |\lambda_n|.$$

Let

$$\begin{aligned} q^{(0)} &= z^{(0)} - (z^{(0)T} x_1) x_1, \\ z^{(r+1)} &= A q^{(r)}, \text{ and} \\ q^{(r+1)} &= z^{(r+1)} - (z^{(r+1)T} x_1) x_1, \end{aligned}$$

where $z^{(0)}$ is randomly chosen. Show that $q^{(r+1)}/\lambda_2^{r+1} \rightarrow c_2 x_2$, where x_2 is the eigenvector corresponding to λ_2 and c_2 is a constant. (Note that the eigenvectors of a symmetric matrix A are orthogonal, that is, $x_j^T x_i = 0$ if $j \neq i$. Also, assume $x_1^T x_1 = 1$.)

9. Apply several iterations of the inverse power method to the matrix from Problem 1 on page 319, using several different starting vectors, and using the centers of the Gerschgorin circles as estimates for λ . Compare the results to the results you obtained from Problem 6 on page 320.

10. Let A be a $(2n+1) \times (2n+1)$ symmetric matrix with elements $a_{ij} = (1.5)^i$ if $i = j$ and $a_{ij} = (0.5)^{i+j-1}$ if $i \neq j$. Let the eigenvalues of A be λ_i , $i = 1, 2, \dots, 2n+1$, ordered such that $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_{2n} \leq \lambda_{2n+1}$. We wish to compute the eigenvector x_{n+1} associated with the middle eigenvalue λ_{n+1} using the inverse power method $q^r = (\lambda I - A)^{-1} q^{r-1}$ for $r = 1, 2, \dots$. Considering Gerschgorin's Theorem for symmetric matrices, choose a value for λ that would ensure rapid convergence. Explain how you chose this value.
11. Compute all of the eigenvalues of the matrix from Problem 1 on page 319, using the power method with deflation. Compare the results to the results you obtained from Problem 1 on page 319.
12. Show that reduction of an n by n matrix to upper Hessenberg form requires $(5/3)n^3 + \mathcal{O}(n^2)$ multiplications.
13. Show that reduction of an n by n Hermitian matrix to upper Hessenberg form can be done with $(2/3)n^3 + \mathcal{O}(n^2)$ multiplications.
14. Show that, in QR iteration (formulas (5.32) and (5.33) on page 308), $A_{\nu+1}$ is upper Hessenberg whenever A_ν is.
15. Prove the assertions in Remark 5.22 on page 309.
16. Let the 3×3 matrix A have eigenvalues $\lambda_1 = 2$, $\lambda_2 = 4$, and $\lambda_3 = 6$, with associated eigenvectors v_1 , v_2 , and v_3 . Consider the iteration method

$$x_{k+1} = -(A + 3I)^{-1}(A - 5I)^{-1}x_k,$$

where $x_0 = v_1 + v_2 + v_3$. Prove that

$$\|x_k - z\| \leq \frac{c}{3^k}$$

for some constant $c > 0$, where z is one of the eigenvectors v_1 , v_2 , or v_3 .

17. Let A be an $n \times n$ matrix and let $b \in \mathbb{R}^n$. Let K be the $n \times n$ matrix $K = (k_1, k_2, \dots, k_n)$ with j -th column $k_j = A^{j-1}b$ for $j = 1, 2, \dots, n$. Assume that K is nonsingular. Let $c = -K^{-1}A^n b$.

(a) Show that

$$AK = K(e_2, e_3, \dots, e_n, -c),$$

where e_i is the i -th column of the identity matrix, so

$$K^{-1}AK = (e_2, e_3, \dots, e_n, -c) = C.$$

(b) Prove that A and C have the same eigenvalues.

(c) Show that

$$\det(C - \lambda I) = (-1)^n \left(\lambda^n + \sum_{i=1}^n c_i \lambda^{i-1} \right) = p(\lambda).$$

(d) Based on (a), (b), and (c), explain why calculation of the eigenvalues of an $n \times n$ matrix with $n \geq 5$ generally cannot be performed in a finite number of steps.

Chapter 6

Numerical Differentiation and Integration

In this chapter, we study the fundamental problem of approximating integrals and derivatives.

6.1 Numerical Differentiation

There are two common ways to develop approximations to derivatives, using Taylor's formula or Lagrange interpolation.

6.1.1 Derivation with Taylor's Theorem

Consider applying Taylor's formula for approximating derivatives. Suppose that $f \in C^2[a, b]$. We wish to approximate $f'(x_0)$ for $x_0 \in (a, b)$. By Taylor's formula,

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{(x - x_0)^2}{2} f''(\xi(x))$$

for some ξ between x and x_0 . Thus, letting $x = x_0 + h$,

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{h}{2} f''(\xi).$$

Hence,

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} + \mathcal{O}(h) \text{ (forward-difference formula)}. \quad (6.1)$$

To obtain a better approximation, suppose that $f \in C^3[a, b]$ and consider

$$\begin{cases} f(x_0 + h) = f(x_0) + f'(x_0)h + f''(x_0)\frac{h^2}{2} + f'''(\xi_1)\frac{h^3}{6} \\ f(x_0 - h) = f(x_0) - f'(x_0)h + f''(x_0)\frac{h^2}{2} - f'''(\xi_2)\frac{h^3}{6}. \end{cases} \quad (6.2)$$

Subtracting the above two expressions and dividing by $2h$ gives

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} + \mathcal{O}(h^2) \text{ (central-difference formula)}. \quad (6.3)$$

Similarly, we can go out one more term in (6.2) (assuming $f \in C^4[a, b]$). Adding the two resulting expressions and dividing by h^2 then gives

$$f''(x_0) = \frac{1}{h^2} [f(x_0 - h) - 2f(x_0) + f(x_0 + h)] - \frac{h^2}{24} [f^{(4)}(\xi_1) + f^{(4)}(\xi_2)].$$

Hence, using the Intermediate Value Theorem,

$$f''(x_0) = \frac{f(x_0 + h) - 2f(x_0) + f(x_0 - h)}{h^2} - \frac{h^2}{12} f^{(4)}(\xi). \quad (6.4)$$

Example 6.1

$f(x) = x \ln x$. Estimate $f''(2)$ using (6.4) with $h = 0.1$. Doing so, we obtain

$$f''(2) \approx \frac{f(2.1) - 2f(2) + f(1.9)}{(0.1)^2} = 0.50021.$$

(Notice that $f''(2) = 1/2$, so the approximation is accurate.) □

6.1.2 Derivation via Lagrange Polynomial Representation

Now consider the Lagrange polynomial method for approximating derivatives. Let $\{x_0, x_1, \dots, x_n\}$ be $n + 1$ distinct points on $[a, b]$ and assume that $x_{j+1} - x_j = h$ (not necessary) for $j = 0, 1, 2, \dots, n - 1$. The Lagrange interpolating polynomial to $f(x)$ of degree at most n through the points $(x_0, f(x_0)), (x_1, f(x_1)), \dots, (x_n, f(x_n))$ is

$$p(x) = \sum_{k=0}^n f(x_k) L_k(x), \text{ where } L_k(x) = \prod_{\substack{i=0 \\ i \neq k}}^n \left(\frac{x - x_i}{x_k - x_i} \right) \quad (6.5)$$

and

$$f(x) = p(x) + \frac{f^{(n+1)}(\xi(x))(x - x_0)(x - x_1) \dots (x - x_n)}{(n + 1)!} \quad (6.6)$$

for some $\xi(x) \in [a, b]$. Hence,

$$\begin{aligned} f'(x) &= \sum_{k=0}^n f(x_k) L'_k(x) + \frac{d}{dx} \left(\prod_{i=0}^n (x - x_i) \right) \frac{f^{(n+1)}(\xi(x))}{(n + 1)!} \\ &\quad + \frac{\prod_{i=0}^n (x - x_i)}{(n + 1)!} \frac{d}{dx} f^{(n+1)}(\xi(x)), \end{aligned}$$

and

$$f'(x_\ell) = \sum_{k=0}^n f(x_k) L'_k(x_\ell) + \left(\prod_{\substack{i=0 \\ i \neq \ell}}^n (x_\ell - x_i) \right) \frac{f^{(n+1)}(\xi(x_\ell))}{(n+1)!} \quad \text{for } x = x_\ell. \quad (6.7)$$

Thus, $f'(x_\ell) \approx \sum_{k=0}^n f(x_k) L'_k(x_\ell)$, since

$$\left(\prod_{\substack{i=0 \\ i \neq \ell}}^n (x_\ell - x_i) \right) \frac{f^{(n+1)}(\xi(x_\ell))}{(n+1)!} \quad \text{is generally small.}$$

Consider, for example, $n = 2$ with $x_1 = x_0 + h, x_2 = x_1 + h$. Then,

$$\begin{aligned} f'(x_\ell) &= f(x_0) \left[\frac{2x_\ell - x_1 - x_2}{(x_0 - x_1)(x_0 - x_2)} \right] + f(x_1) \left[\frac{2x_\ell - x_0 - x_2}{(x_1 - x_0)(x_1 - x_2)} \right] \\ &\quad + f(x_2) \left[\frac{2x_\ell - x_0 - x_1}{(x_2 - x_0)(x_2 - x_1)} \right] + \frac{1}{6} f'''(\xi_\ell) \prod_{\substack{i=0 \\ i \neq \ell}}^2 (x_\ell - x_i). \end{aligned}$$

Taking $x_\ell = x_1$ gives

$$\begin{aligned} f'(x_1) &= -\frac{1}{2h} f(x_0) + 0 f(x_1) + \frac{1}{2h} f(x_2) - \frac{1}{6} f'''(\xi_1) h^2 \\ &= \frac{1}{2h} [f(x_1 + h) - f(x_1 - h)] - \frac{1}{6} f'''(\xi_1) h^2 \quad (\text{central difference formula}). \end{aligned}$$

By considering 5 points ($n = 4$), derivative formulas of $\mathcal{O}(h^4)$ can be derived. In particular,

$$f'(x_0) = \frac{1}{12h} [f(x_0 - 2h) - 8f(x_0 - h) \quad (6.8)$$

$$+ 8f(x_0 + h) - f(x_0 + 2h)] + \frac{h^4}{30} f^{(5)}(\xi)$$

$$\begin{aligned} f'(x_0) &= \frac{1}{12h} [-25f(x_0) + 48f(x_0 + h) - 36f(x_0 + 2h) \\ &\quad + 16f(x_0 + 3h) - 3f(x_0 + 4h)] + \frac{h^4}{5} f^{(5)}(\xi). \end{aligned} \quad (6.9)$$

6.1.3 Error Analysis

One difficulty with numerical differentiation is that rounding error can be large if h is too small. In the computer, $f(x_0 + h) = \tilde{f}(x_0 + h) + e(x_0 + h)$ and $f(x_0) = \tilde{f}(x_0) + e(x_0)$, where $e(x_0 + h)$ and $e(x_0)$ are roundoff errors that depend on the number of digits used by the computer.

Consider the forward-difference formula

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} + \frac{h}{2}f''(\xi(x)).$$

We will assume that $|e(x)| \leq \epsilon|f(x)|$ for some relative error ϵ , that $|f(x)| \leq M_0$ for some constant M_0 , and that $|f''(x)| \leq M_2$ for some constant M_2 , for all values of x near x_0 that are being considered. Then, these assumed bounds and repeated application of the triangle inequality give

$$\begin{aligned} \left| f'(x_0) - \frac{\tilde{f}(x_0 + h) - \tilde{f}(x_0)}{h} \right| &\leq \left| \frac{e(x_0 + h) - e(x_0)}{h} \right| + \frac{h}{2}M_2 \\ &\leq \frac{2\epsilon M_0}{h} + \frac{hM_2}{2} = E(h), \end{aligned} \quad (6.10)$$

where ϵ is any number such that $|e(x)| \leq \epsilon|f(x)|$ for all x under consideration. That is, the error is bounded by a curve such as that in Figure 6.1. Thus, if

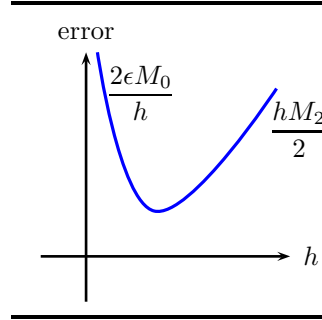


FIGURE 6.1: Illustration of the total error (roundoff plus truncation) bound in forward difference quotient approximation to f' .

the value of h is too small, the error can be large.

Example 6.2

(Using 10-digit precision) $f(x) = \ln x$, $\frac{1}{3} = f'(3) \approx \frac{\ln(3+h) - \ln(3)}{h}$.

h	$\frac{\ln(3+h)-\ln(3)}{h}$
10^{-1}	0.32790
10^{-2}	0.33278
10^{-3}	0.33328
10^{-4}	0.33332 best estimate
10^{-5}	0.33330
10^{-6}	0.33300
10^{-7}	0.33000
10^{-8}	0.30000
10^{-9}	0

To analyze this example, notice that $f''(x) = -\frac{1}{x^2}$ and $M_2 = \max |f''(\xi)| \approx \frac{1}{9}$. Suppose that the error is $\frac{2\epsilon}{h}M_0 + \frac{h}{2}M_2$, so $e(h) = \frac{2\epsilon}{h}M_0 + \frac{h}{2}M_2$. The minimum error occurs at $e'(h) = 0$, which gives $h_{\text{opt}} \approx \sqrt{36\epsilon}$. For ten significant digit accuracy, $\epsilon \approx 5 \times 10^{-10}$. Thus, $h_{\text{opt}} \approx 10^{-4}$. $\square$

If we use calculus to minimize the expression on the right of (6.10) with respect to h , we obtain

$$h_{\text{opt}} = \frac{2\sqrt{\epsilon M_0}}{\sqrt{M_2}},$$

with a minimal bound on the error of

$$E(h_{\text{opt}}) = 2\sqrt{M_0 M_2} \sqrt{\epsilon}.$$

Although the right member of (6.10) is merely a bound, we see that h_{opt} gives a good estimate for the optimal step in the divided difference, and $E(h_{\text{opt}})$ gives a good estimate for the minimum achievable error. In particular, the minimum achievable error is $\mathcal{O}(\sqrt{\epsilon})$ and the optimal h is also $\mathcal{O}(\sqrt{\epsilon})$, both in the estimates and in the numerical experiments in Example 6.2.

With higher-order formulas, we can obtain a smaller total error bound, at the expense of additional complication. In particular, if the roundoff error is $\mathcal{O}(1/h)$ and the truncation error is $\mathcal{O}(h^n)$, then the optimal h is $\mathcal{O}(\epsilon^{1/(n+1)})$ and the minimum achievable error bound is $\mathcal{O}(\epsilon^{n/(n+1)})$.

6.2 Automatic (Computational) Differentiation

Numerical differentiation has been used extensively in the past, e.g. for computing the derivative f' for use in Newton's method.¹ Another example of the use of such derivative formulas is in the construction of methods for

¹or the multidimensional analog, as described in §8.1 on page 439

the solution of boundary value problems in differential equations, such as is explained in §10.1.2, starting on page 540 below. However, as we have just seen (in §6.1.3 above) roundoff error limits the accuracy of finite-difference approximations to derivatives. Moreover, it may be difficult in practice to determine a step size h for which near-optimal accuracy can be attained. This can cause significant problems, for example, in multivariate floating point Newton methods.

For complicated functions, algebraic computation of the derivatives by hand is also impractical. One possible alternative is to compute the derivatives with symbolic manipulation systems such as Mathematica, Maple, or Reduce. These systems have facilities for output of the derivatives as statements in common compiled programming languages. However, such systems are often not able to adequately simplify the expressions for the derivatives, resulting in expressions for derivatives that can be many times as long as the expressions for the function itself. This “expression swell” not only can result in inefficient evaluation, but also can cause roundoff error to be a problem, even though there is no truncation error.

A third alternative is *automatic differentiation*, also called “computational differentiation.” In this scheme, there is no truncation (method) error and the expression for the function is not symbolically manipulated, yet the user only need supply the expression for the function itself. The technique, increasingly used during the two decades prior to composition of this book, is based upon defining an arithmetic on composite objects, the components of which represent function and derivative values. The rules of this arithmetic are based on the elementary rules of differentiation learned in calculus, in particular, on the chain rule.

6.2.1 The Forward Mode

In the “forward mode” of automatic differentiation, the derivative or derivatives are computed at the same time as the function. For example, if the function and the first k derivatives are desired, then the arithmetic will operate on objects of the form

$$u_{\nabla} = \langle u, u', u'', \dots, u^{(k)} \rangle. \quad (6.11)$$

Addition of such objects comes from the calculus rule “the derivative of a sum is the sum of the derivatives,” that is,

$$u_{\nabla} + v_{\nabla} = \langle u + v, u' + v', u'' + v'', \dots, u^{(k)} + v^{(k)} \rangle. \quad (6.12)$$

In other words, the j -th component of $u_{\nabla} + v_{\nabla}$ is the j -th component of u_{∇} plus the j -th component of v_{∇} , for $1 \leq j \leq k$. Subtraction is defined similarly, while products $u_{\nabla} v_{\nabla}$ are defined such that the first component of $u_{\nabla} v_{\nabla}$ is

the first component of u_{∇} times the first component of v_{∇} , etc., as follows:

$$u_{\nabla}v_{\nabla} = \left\langle uv, u'v + uv', u''v + 2u'v' + uv'', \dots, \sum_{j=0}^k \binom{k}{j} u^{(k-j)}v^{(j)} \right\rangle. \quad (6.13)$$

Rules for applying functions such as “exp,” “sin,” and “cos” to such objects are similarly defined. For example,

$$\sin(u_{\nabla}) = \langle \sin(u), u' \cos(u), -\sin(u)(u')^2 + \cos(u)u'', \dots \rangle. \quad (6.14)$$

The differentiation object corresponding to a particular value a of the independent variable x is of the form

$$x_{\nabla} = \langle a, 1, 0, \dots, 0 \rangle.$$

Example 6.3

Suppose the context requires us to have values of the function, of the first derivative, and of the second derivative for the function

$$f(x) = x \sin(x) - 1,$$

where we want function and derivative values at $x = \pi/4$. What steps would the computer do to complete the automatic differentiation?

The computer would first resolve f into a sequence of operations (sometimes called a *code list*, *tape*, or *decomposition into elementary operations*). If we associate the independent variable x with the variable v_1 and the i -th intermediate result with v_{i+1} , a sequence of operations for f can be²

$$\begin{aligned} v_{\nabla 2} &\leftarrow \sin(v_{\nabla 1}) \\ v_{\nabla 3} &\leftarrow v_{\nabla 1}v_{\nabla 2} \\ v_{\nabla 4} &\leftarrow v_{\nabla 3} - 1 \end{aligned} \quad (6.15)$$

We now illustrate with 4-digit decimal arithmetic, with rounding to nearest. We first set

$$v_{\nabla 1} \leftarrow \langle \pi/4, 1, 0 \rangle \approx \langle 0.7854, 1, 0 \rangle.$$

Second, we use (6.14) to obtain

$$\begin{aligned} v_{\nabla 2} &\leftarrow \sin(\langle 0.7854, 1, 0 \rangle) \\ \text{i.e. } &\langle \sin(0.7854), 1 \times \cos(0.7854), -\sin(0.7854) \times (1^2) + \cos(0.7854) \times 0 \rangle \\ &\approx \langle 0.7071, 0.7071, -0.7071 \rangle. \end{aligned}$$

²We say “a sequence of operations for f can be,” rather than “the sequence of operations for f is,” because, in general, decompositions for a particular expression are not unique.

Third, we use (6.13) to obtain

$$\begin{aligned} v_{\nabla_3} &\leftarrow \langle 0.7854, 1, 0 \rangle \langle 0.7071, 0.7071, -0.7071 \rangle \\ \text{i.e. } &\langle 0.7854 \times 0.7071, 1 \times 0.7071 + 0.7854 \times 0.7071, \\ &0 \times 0.7071 + 2 \times 1 \times 0.7071 + 0.7854 \times (-0.7071) \rangle \\ &\approx \langle 0.5554, 1.263, 0.8589 \rangle \end{aligned}$$

Finally, the second derivative object corresponding to the constant 1 is $\langle 1, 0, 0 \rangle$, so we apply formula (6.12) to obtain

$$\begin{aligned} v_{\nabla_4} &\leftarrow \langle 0.5554, 1.263, 0.8589 \rangle - \langle 1, 0, 0 \rangle \\ &\approx \langle -0.4446, 1.263, .08589 \rangle. \end{aligned}$$

Comparing, we have

$$\begin{aligned} f(\pi/4) &= (\pi/4 \sin(\pi/4) - 1 \approx -0.4446, \\ f'(x) &= x \cos(x) + \sin(x) \quad \text{so} \quad f'(\pi/4) \approx 1.262, \\ f''(x) &= -x \sin(x) + 2 \cos(x) \quad \text{so} \quad f''(\pi/4) \approx 0.8589, \end{aligned}$$

where the above values were computed to 16 digits, then rounded to four digits. This illustrates the validity of automatic differentiation.³ $\square$

6.2.2 The Reverse Mode

The reverse mode of automatic differentiation, when used to compute the gradient of a function f of n variables, can be more efficient than the forward mode. In particular, when the forward mode (or for that matter, when finite differences or when symbolic derivatives) is used, the number of operations required to compute the gradient is proportional to n times the number of operations to compute the function. In contrast, when the reverse mode is used, it can be proven that the number of operations required to compute the the gradient ∇F (which has n components) is bounded by 5 times the number of operations required to evaluate the f itself, regardless of n . (However, a quantity of numbers proportional to the number of operations required to evaluate f needs to be stored when the reverse mode is used.)

So, how does the reverse mode work? We can think of the reverse mode as forming a system of equations relating the derivatives of the intermediate variables in the computation through the chain rule, then solving the system of equations for the derivative of the independent variable. Suppose we have

³The discrepancy between the values 1.263 and 1.262 for $f'(\pi/4)$ is due to the fact that rounding to four digits was done after each operation in the automatic differentiation. If the expression for f' were first symbolically derived, then evaluated with four digit rounding (rather than exactly, then rounding), then a similar error would occur.

a code list such as (6.15), giving the sequence of instructions for evaluating a function f . For example, one such operation could be

$$v_p = v_q + v_r,$$

where v_p is the value to be computed, while v_q and v_r have previously been computed. Then, computing f' is equivalent to computing v'_M , where v_M corresponds to the value of f . (That is, v_M is the dependent variable, generally the result of the last operation in the computation of the expression for f .) We form a sparse linear system with an equation for each operation in the code list, whose variables are v'_k , $1 \leq k \leq M$. For example, the equation corresponding to an addition $v_p = v_q + v_r$ would be

$$v'_q + v'_r - v'_p = 0,$$

while the equation corresponding to a product $v_p = v_q v_r$ would be

$$v_r v'_q + v_q v'_r - v'_p = 0,$$

where the values of the intermediate quantities v_q and v_r have been previously computed and stored from an evaluation of f . Likewise, if the operation were $v_p = \sin(v_q)$, then the equation would be

$$\cos(v_q) v'_q - v'_p = 0,$$

while if the operation were addition of a constant, $v_p = v_q + c$, then the equation would be

$$v'_q - v'_p = 0.$$

If there is a single independent variable and the derivative is with respect to this variable, then the first equation would be

$$v'_1 = 1.$$

We illustrate with the f for Example 6.3. If the code list is as in (6.15), then the system of equations will be

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ \cos(v_1) & -1 & 0 & 0 \\ v_2 & v_1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} v'_1 \\ v'_2 \\ v'_3 \\ v'_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \quad (6.16)$$

If $v_1 = x = \pi/4$ as in Example 6.3, then this system, filled using four-digit arithmetic, is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.7071 & -1 & 0 & 0 \\ 0.7071 & 0.7854 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} v'_1 \\ v'_2 \\ v'_3 \\ v'_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \quad (6.17)$$

The reverse mode consists simply of solving this system with forward substitution. This system has solution

$$\begin{pmatrix} v'_1 \\ v'_2 \\ v'_3 \\ v'_4 \end{pmatrix} \approx \begin{pmatrix} 1.0000 \\ 0.7071 \\ 1.2625 \\ 1.2625 \end{pmatrix}.$$

Thus $f'(\pi/4) = v'_4 \approx 1.2625$, which corresponds to what we obtained with the forward mode.

Example 6.4

Suppose $f(x_1, x_2) = x_1^2 - x_2^2$. Compute

$$\nabla f(x_1, x_2) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \right)^T$$

at $(x_1, x_2) = (1, 2)$ using the reverse mode.

Solution: A code list for this function can be

$$\begin{aligned} v_1 &= x_1 \\ v_2 &= x_2 \\ v_3 &= v_1^2 \\ v_4 &= v_2^2 \\ v_5 &= v_3 - v_4 \end{aligned}$$

The reverse mode system of equations for computing df/dx_i is thus

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 2v_1 & 0 & -1 & 0 & 0 \\ 0 & 2v_2 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & -1 \end{pmatrix} \begin{pmatrix} v'_1 \\ v'_2 \\ v'_3 \\ v'_4 \\ v'_5 \end{pmatrix} = e_i, \quad (6.18)$$

where e_i is the vector whose i -th component is 1 and all of whose other components are 0. When $x_1 = 1$ and $x_2 = 2$, we have

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & -1 & 0 & 0 \\ 0 & 4 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & -1 \end{pmatrix} \begin{pmatrix} v'_1 \\ v'_2 \\ v'_3 \\ v'_4 \\ v'_5 \end{pmatrix} = e_i.$$

Now, df/dx_1 can be computed by ignoring the row and column corresponding to v'_2 , while df/dx_2 can be computed by ignoring the row and column corresponding to v'_1 . We thus obtain $\partial f/\partial x_1 = 2$ and $\partial f/\partial x_2 = -4$ (Exercise 9).

□

In fact, a directional derivative can be computed in the reverse mode with the same amount of work it takes to compute a single partial derivative. For example, the directional derivative of $f(x_1, x_2)$ at $(x_1, x_2) = (1, 2)$ in the direction of $u = (1/\sqrt{2}, 1/\sqrt{2})^T$ can be obtained by solving the linear system

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & -1 & 0 & 0 \\ 0 & 4 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & -1 \end{pmatrix} \begin{pmatrix} v'_1 \\ v'_2 \\ v'_3 \\ v'_4 \\ v'_5 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (6.19)$$

for v'_5 .

6.2.3 Implementation of Automatic Differentiation

Automatic differentiation can be incorporated directly into the programming language compiler, or the technology of *operator overloading* (available in object-oriented languages) can be used. A number of packages are available to do automatic differentiation. The best packages (such as **ADOLC**, for differentiating “C” programs and **ADIFOR**, for differentiating Fortran programs) can accept the definition of the function f in the form of a fairly generally written computer program. Some of them (such as **ADIFOR**) produce a new program that will evaluate both the function and derivatives, while others (such as **ADOLC**) produce a code list or “tape” from the original program, then operate on the code list to produce the derivatives. The monograph [36] contains a comprehensive overview of theory and implementation of both the forward and backward modes.

6.3 Numerical Integration

The problem throughout the remainder of this chapter is determining accurate methods for approximating the integral $\int_a^b f(x)dx$. Approximating integrals is called *numerical integration* or *quadrature*.

6.3.1 Introduction

First, we define some useful notation. Let

$$J(f) = \int_a^b f(x)dx. \quad (6.20)$$

We will see that most quadrature formulas reduce to the form

$$Q(f) = (b - a) \sum_{j=0}^m \alpha_j f(x_j), \quad (6.21)$$

where the $\alpha_0, \alpha_1, \dots, \alpha_m$ are called *weights* and the $x_0, x_1, \dots, x_m$ are the *sample or nodal points*. We have

$$J(f) = Q(f) + E(f), \quad (6.22)$$

where $E(f)$ is the error in the quadrature formula.

REMARK 6.1 To obtain $E(f) = 0$ for $f(x) = a$ constant, we require $\sum_{j=0}^m \alpha_j = 1$. □

In the next few sections, we will study quadrature formulas for which the error $E(f)$ is proportional to $H^{g(m)}$, where $g(m) > 1$ depends on the weights and sample points and $H = b - a$. For fixed m , which is generally desirable since the weights and points may be either difficult to compute or the weights may become large in magnitude,⁴ we need a way to make $E(f)$ go to zero. This is accomplished by dividing the interval $[a, b]$ into many smaller subintervals. This procedure is called *composite numerical integration*.

Consider Figure 6.2. We divide $[a, b]$ into N subintervals each of length h .

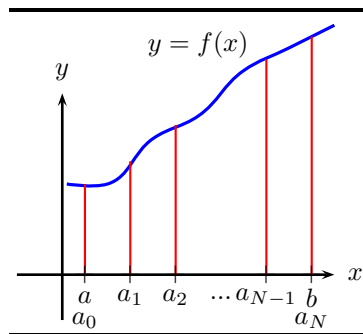


FIGURE 6.2: Illustration of composite numerical integration.

⁴leading to rounding errors

Thus, $h = (b - a)/N$ and $a_\nu = a + \nu h$ for $\nu = 0, 1, \dots, N$. Then,

$$J(f) = \sum_{\nu=0}^{N-1} \int_{a_\nu}^{a_{\nu+1}} f(x) dx = \sum_{\nu=0}^{N-1} [Q_\nu(f) + E_\nu(f)], \quad (6.23)$$

where $Q_\nu(f) = (a_{\nu+1} - a_\nu) \sum_{j=0}^m \alpha_j f(x_{\nu,j})$ and $E_\nu(f) = \mathcal{O}(h^{g(m)})$.

Thus,

$$\sum_{\nu=0}^{N-1} Q_\nu(f) = J(f) + \mathcal{O}(Nh^{g(m)}) = J(f) + \mathcal{O}(h^{g(m)-1}) \rightarrow J(f) \text{ as } h \rightarrow 0.$$

REMARK 6.2 In practice, the points and weights, $x_j, \alpha_j, j = 0, 1, \dots, m$, are generally given for some interval, say $[-1, 1]$. Then

$$\int_{-1}^1 f(x) dx \approx 2 \sum_{j=0}^m \alpha_j f(x_j).$$

However, given these points, the points for any interval $[a_\nu, a_{\nu+1}]$ can be determined, and hence

$$Q_\nu(f) \approx \int_{a_\nu}^{a_{\nu+1}} f(x) dx$$

can be calculated. To see how, first note that

$$\int_{a_\nu}^{a_{\nu+1}} f(x) dx = \int_{-1}^1 f\left(\frac{zh + a_{\nu+1} + a_\nu}{2}\right) \frac{h}{2} dz,$$

where

$$x = \frac{zh + a_{\nu+1} + a_\nu}{2} \text{ and } z = \frac{2x - a_{\nu+1} - a_\nu}{h}.$$

Thus,

$$\begin{aligned} \int_{a_\nu}^{a_{\nu+1}} f(x) dx &\approx Q_\nu(f) = h \sum_{j=0}^m f\left(\frac{x_j h + a_{\nu+1} + a_\nu}{2}\right) \alpha_j \\ &= h \sum_{j=0}^m f\left(\frac{hx_j + 2a + (2\nu + 1)h}{2}\right) \alpha_j. \end{aligned}$$

Therefore,

$$J(f) = \sum_{\nu=0}^{N-1} h \sum_{j=0}^m \alpha_j f(x_{\nu,j}) + \mathcal{O}(h^{g(m)-1}), \quad (6.24)$$

where $x_{\nu,j} = (hx_j + 2a + (2\nu + 1)h)/2$ for $0 \leq j \leq m$, $0 \leq \nu \leq N - 1$. $\square$

In the next two sections, we will study two popular ways of determining accurate points and weights, x_j and α_j for $j = 0, 1, \dots, m$, to be used in formulas (6.21) and (6.24).

6.3.2 Newton-Cotes Formulas

Consider

$$J(f) = \int_a^b f(x)dx \approx Q(f) = (b-a) \sum_{j=0}^m \alpha_j f(x_j).$$

We now study a standard procedure for determining the α_j 's and x_j 's to obtain approximations called the *Newton-Cotes Formulas*. (Of course, these weights and points are usually implemented in the composite formula (6.24).) There are two ways to derive the Newton-Cotes Formulas; each is useful to understand. For either derivation, the points x_j , $j = 0, 1, \dots, m$ are taken to be equally spaced on the interval $[a, b]$. There are two ways to accomplish this.

DEFINITION 6.1 *The $(m+1)$ point open Newton-Cotes formulas have points $x_j = x_0 + jh$, $j = 0, 1, 2, \dots, m$, where $h = (b-a)/(m+2)$ and $x_0 = a + h$. The $(m+1)$ point closed Newton-Cotes formulas have points $x_j = x_0 + jh$, $j = 0, 1, \dots, m$, where $h = (b-a)/(m)$ and $x_0 = a$.*

REMARK 6.3 The h in Definition 6.1 is different from the h used in the composite numerical integration method. $\square$

Example 6.5

Suppose that $a = 0$, $b = 1$, and $m = 3$. Then, the open points are $x_0 = 0.2$, $x_1 = 0.4$, $x_2 = 0.6$, and $x_3 = 0.8$, while the closed points are $x_0 = 0$, $x_1 = 1/3$, $x_2 = 2/3$, and $x_3 = 1$. $\square$

We now consider two ways for deriving the weights α_j .

6.3.2.1 Derivation 1 (Require (6.21) to be exact for polynomials of degree $\leq m$)

We require

$$J(p) = Q(p) \text{ for polynomials } p(x) \text{ up to degree } m. \quad (6.25)$$

Consider

$$\begin{aligned}
 E(f) &= J(f) - Q(f) \\
 &= J(f) - J(p) + Q(p) - Q(f) \text{ for any } p \in P_m \\
 &= J(f - p) + Q(p - f) \\
 &= \int_a^b (f(x) - p(x)) dx + (b - a) \sum_{j=0}^m \alpha_j (p(x_j) - f(x_j)).
 \end{aligned}$$

Thus,

$$|E(f)| \leq ((b - a) + (b - a) \sum_{j=0}^m |\alpha_j|) \max_{a \leq x \leq b} |f(x) - p(x)|$$

for any $p \in P_m$. Hence, if $J(p) = Q(p)$ and f is smooth, our polynomial approximation theory results⁵ indicate that the error $|E(f)|$ may be quite small.

We now have the question: Does condition (6.25) uniquely determine the weights α_j ? Consider first the closed Newton–Cotes formulas for $m = 3$ with $a = 0$ and $b = 1$.

Then,

$$Q(f) = \alpha_0 f(0) + \alpha_1 f(1/3) + \alpha_2 f(2/3) + \alpha_3 f(1)$$

is exact for $f(x) = 1, x, x^2, x^3$. This produces the linear system

$$\begin{aligned}
 \int_0^1 1 \, dx = 1 &= \alpha_0 + \alpha_1 + \alpha_2 + \alpha_3, \\
 \int_0^1 x \, dx = \frac{1}{2} &= \alpha_0(0) + \alpha_1\left(\frac{1}{3}\right) + \alpha_2\left(\frac{2}{3}\right) + \alpha_3(1), \\
 \int_0^1 x^2 \, dx = \frac{1}{3} &= \alpha_0(0) + \alpha_1\left(\frac{1}{3}\right)^2 + \alpha_2\left(\frac{2}{3}\right)^2 + \alpha_3(1)^2, \\
 \int_0^1 x^3 \, dx = \frac{1}{4} &= \alpha_0(0) + \alpha_1\left(\frac{1}{3}\right)^3 + \alpha_2\left(\frac{2}{3}\right)^3 + \alpha_3(1)^3.
 \end{aligned}$$

Solving we obtain $\alpha_0 = 1/8, \alpha_1 = 3/8, \alpha_2 = 3/8, \alpha_3 = 1/8$.

In general, we obtain the linear system $A\alpha = b$ where

$$A = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_0 & x_1 & x_2 & \cdots & x_m \\ x_0^2 & x_1^2 & x_2^2 & \cdots & x_m^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_0^m & x_1^m & x_2^m & \cdots & x_m^m \end{pmatrix}, \quad \alpha = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{pmatrix}, \quad \text{and} \quad b = \begin{pmatrix} 1 \\ 1/2 \\ 1/3 \\ \vdots \\ 1/(m+1) \end{pmatrix}$$

⁵e.g., the error terms for Lagrange interpolation or best uniform approximation

for the interval $[0, 1]$. Note that A is the transpose of the *Vandermonde matrix* that arises as the matrix for the system of equations for the coefficients of the interpolating polynomial at the points x_i . (See page 211.) Thus, the matrix A is nonsingular, because $x_j \neq x_\ell$ for $j \neq \ell$.

6.3.2.2 Derivation 2 (Require (6.21) to be exact for the Lagrange interpolant to $f(x)$)

Recall Lagrange polynomial interpolation

$$f(x) = p(x) + R(x),$$

where

$$p(x) = \sum_{j=0}^m f(x_j) L_j(x) \quad \text{and} \quad R(x) = \frac{f^{m+1}(\xi(x))}{(m+1)!} \prod_{i=0}^m (x - x_i),$$

$a \leq \xi(x) \leq b$, and

$$L_j(x) = \prod_{\substack{k=0 \\ k \neq j}}^m \frac{(x - x_k)}{(x_j - x_k)}.$$

Recall that $f(x_i) = p(x_i)$ for $i = 0, 1, \dots, m$, where the x_i are either the open or closed points. Consider

$$\begin{aligned} J(f) &= \int_a^b f(x) dx = \int_a^b p(x) dx + \int_a^b R(x) dx \\ &= \sum_{j=0}^m f(x_j) \int_a^b L_j(x) dx + \int_a^b R(x) dx \\ &= \sum_{j=0}^m f(x_j) \alpha_j + E(f). \end{aligned}$$

Define

$$Q(f) = \sum_{j=0}^m \alpha_j f(x_j), \quad \text{where} \quad \alpha_j = \int_a^b L_j(x) dx \quad \text{and} \quad E(f) = \int_a^b R(x) dx.$$

Example 6.6

Consider the closed Newton–Cotes formula on $[0, 1]$ for $m = 3$. In this case, $x_0 = 0$, $x_1 = 1/3$, $x_2 = 2/3$, $x_3 = 1$. Then,

$$\begin{aligned} \alpha_0 &= \int_0^1 L_0(x) dx = \int_0^1 \frac{(x - x_1)(x - x_2)(x - x_3)}{(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)} dx \\ &= -\frac{9}{2} \int_0^1 \left(x^3 - 2x^2 + \frac{11}{9}x - \frac{2}{9} \right) dx = \frac{1}{8}. \end{aligned}$$

Similarly, $\alpha_1 = 3/8$, $\alpha_2 = 3/8$, $\alpha_3 = 1/8$. Thus,

$$\int_0^1 f(x)dx \approx \frac{1}{8}f(0) + \frac{3}{8}f\left(\frac{1}{3}\right) + \frac{3}{8}f\left(\frac{2}{3}\right) + \frac{1}{8}f(1).$$

As an example of application of this formula, consider

$$\int_0^1 e^x dx = e^1 - 1 \approx \frac{1}{8}e^0 + \frac{3}{8}e^{1/3} + \frac{3}{8}e^{2/3} + \frac{1}{8}e^1 \approx 1.71854.$$

(Of course, composite numerical integration can be used with this 4-point Newton Cotes method to improve the approximation.) $\square$

6.3.2.3 The Error Term

We now consider $E(f)$ in greater detail. Recall that

$$E(f) = \int_a^b R(x)dx = \int_a^b \prod_{i=0}^m (x - x_i) \frac{f^{(m+1)}(\xi(x))}{(m+1)!} dx.$$

Case 1 $\prod_{i=0}^m (x - x_i)$ does or does not change sign on $[a, b]$.

Then,

$$\begin{aligned} |E(f)| &= \left| \int_a^b \prod_{i=0}^m (x - x_i) \frac{f^{(m+1)}(\xi(x))}{(m+1)!} dx \right| \\ &\leq \frac{\|f^{(m+1)}\|_\infty}{(m+1)!} \int_a^b \left| \prod_{i=0}^m (x - x_i) \right| dx, \end{aligned}$$

where

$$\|f^{(m+1)}\|_\infty = \max_{a \leq x \leq b} |f^{(m+1)}(x)|.$$

Changing variables,

$$z = \frac{x-a}{b-a}, \quad z_i = \frac{x_i-a}{b-a}, \quad (x - x_i) = (z - z_i)(b-a),$$

gives

$$|E(f)| \leq \frac{\|f^{(m+1)}\|_\infty}{(m+1)!} (b-a)^{m+2} \int_0^1 \left| \prod_{i=0}^m (z - z_i) \right| dz.$$

Thus,

$$|E(f)| \leq H^{m+2} \beta_{m+1}^* \frac{\|f^{(m+1)}\|_\infty}{(m+1)!}, \quad (6.26)$$

where

$$H = (b-a) \text{ and } \beta_{m+1}^* = \int_0^1 \left| \prod_{i=0}^m (z - z_i) \right| dz.$$

Case 2 $\prod_{i=0}^m (x - x_i)$ does not change sign on $[a, b]$ (restrictive)

Then,

$$E(f) = \frac{f^{(m+1)}(\xi')}{(m+1)!} \int_a^b \prod_{i=0}^m (x - x_i) dx,$$

for some $\xi', a \leq \xi' \leq b$, by the weighted mean-value theorem for integrals. Thus,

$$E(f) = \frac{f^{(m+1)}(\xi')}{(m+1)!} (b-a)^{m+2} \int_0^1 \prod_{i=0}^m (z - z_i) dz$$

and

$$E(f) = H^{m+2} \frac{f^{(m+1)}(\xi')}{(m+1)!} \beta_{m+1}, \quad (6.27)$$

where

$$\beta_{m+1} = \int_0^1 \prod_{i=0}^m (z - z_i) dz.$$

Example 6.7

Consider $m = 1$, $x_0 = a$, $x_1 = b$, and closed Newton–Cotes. Then

$$\begin{aligned} J(f) &= \int_a^b f(x) dx = Q(f) + E(f) \\ &= \sum_{j=0}^m f(x_j) \int_a^b L_j(x) dx + \int_a^b (x-a)(x-b) \frac{f''(\xi)}{2!} dx. \end{aligned}$$

Thus,

$$Q(f) = f(a) \int_a^b \frac{x-b}{a-b} dx + f(b) \int_a^b \frac{x-a}{b-a} dx = (f(a) + f(b)) \frac{b-a}{2}$$

(the trapezoidal rule). Also,

$$E(f) = \frac{f''(\xi')}{2} \int_a^b (x-a)(x-b) dx,$$

since $(x-a)(x-b) \leq 0$ for $a \leq x \leq b$.

Thus,

$$E(f) = -\frac{1}{12} (b-a)^3 f''(\xi').$$

Hence,

$$\int_a^b f(x) dx = \frac{b-a}{2} [f(a) + f(b)] - \frac{1}{12} (b-a)^3 f''(\xi').$$

□

Example 6.8

Consider $m = 0$, $x_0 = (a+b)/2$, and open Newton–Cotes. Using the Lagrange polynomial procedure, the error can be shown to be proportional to H^2 for this rule. However, a better result can be obtained by expanding $f(x)$ in a Taylor series about x_0 :

$$f(x) = f\left(\frac{a+b}{2}\right) + f'\left(\frac{a+b}{2}\right)\left(x - \frac{a+b}{2}\right) + f''(\xi(x))\frac{(x - (a+b)/2)^2}{2}.$$

Thus,

$$\int_a^b f(x)dx = (b-a)f\left(\frac{a+b}{2}\right) + 0 + \frac{1}{2}\int_a^b f''(\xi(x))\left(x - \frac{a+b}{2}\right)^2 dx.$$

Hence,

$$\begin{aligned} J(f) &= \int_a^b f(x)dx = Q(f) + \frac{1}{2}f''(\xi^*)\int_a^b \left(x - \frac{a+b}{2}\right)^2 dx \\ &= Q(f) + \frac{1}{24}(b-a)^3 f''(\xi^*) \end{aligned}$$

for some ξ^* , $a \leq \xi^* \leq b$, where

$$Q(f) = (b-a)f\left(\frac{a+b}{2}\right) \quad (\text{the midpoint rule}). \quad (6.28)$$

Therefore, for this rule,

$$E(f) = \frac{1}{24}H^3 f''(\xi^*).$$

□

REMARK 6.4 As indicated in Example 6.8 above, the error analysis starting on page 339 can be improved for certain Newton–Cotes formulas. Indeed, the following results can be derived:

(i) For closed Newton–Cotes formulas:

(a) (m even)

$$E(f) = \frac{H^{m+3}f^{(m+2)}(\xi)}{m^{m+3}(m+2)!} \int_0^m t^2(t-1)\dots(t-m)dt.$$

(b) (m odd)

$$E(f) = \frac{H^{m+2}f^{(m+1)}(\xi)}{m^{m+2}(m+1)!} \int_0^m t(t-1)\dots(t-m)dt.$$

(ii) For open Newton–Cotes formulas:

(a) (m even)

$$E(f) = \frac{H^{m+3} f^{(m+2)}(\xi)}{(m+2)^{m+3} (m+2)!} \int_{-1}^{m+1} t^2 (t-1) \dots (t-m) dt.$$

(b) (m odd)

$$E(f) = \frac{H^{m+2} f^{(m+1)}(\xi)}{(m+2)^{m+2} (m+1)!} \int_{-1}^{m+1} t(t-1) \dots (t-m) dt.$$

for some ξ , $a \leq \xi \leq b$ in all the above formulas. $\square$

REMARK 6.5 The degree of precision of a quadrature formula is defined as the positive integer n satisfying $E(p_k) = 0$ for $k = 0, 1, \dots, n$, where p_k is a polynomial of degree k , but $E(p_{n+1}) \neq 0$ for some polynomial of degree $(n+1)$. By the previous remark, the degree of precision of the Newton–Cotes formulas is $(m+1)$ if m is even and m if m is odd. $\square$

REMARK 6.6 Some common open and closed Newton–Cotes formulas are the following:

(i) *Closed Newton–Cotes*

$m = 1$: (trapezoidal rule)

$$\int_a^b f(x) dx = \frac{b-a}{2} [f(a) + f(b)] - \frac{1}{12} H^3 f''(\xi).$$

$m = 2$: (Simpson's rule)

$$\int_a^b f(x) dx = \frac{b-a}{6} \left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right] - \frac{1}{2880} H^5 f^{(4)}(\xi).$$

(ii) *Open Newton–Cotes*

$m = 0$: (midpoint rule)

$$\int_a^b f(x) dx = (b-a) f\left(\frac{a+b}{2}\right) + \frac{H^3}{24} f''(\xi).$$

$m = 1$: (two-point open rule)

$$\begin{aligned} \int_a^b f(x) dx &= \frac{(b-a)}{2} \left[f\left(a + \frac{b-a}{3}\right) + f\left(a + 2\left(\frac{b-a}{3}\right)\right) \right] \\ &\quad + \frac{H^3}{36} f''(\xi). \end{aligned}$$



REMARK 6.7 It is possible to find $f \in C^\infty[a, b]$ such that $E(f) = J(f) - Q(f)$ does not go to zero as $m \rightarrow \infty$. (See [22].) Thus, composite methods are required, in general, if Newton–Cotes formulas are used. (You will show that $E(f) \rightarrow 0$ as $h \rightarrow 0$ for composite quadrature even for $f \in C[a, b]$; see Exercise 11 on page 377.) □

6.3.3 Gaussian Quadrature

We have just seen that Newton–Cotes formulas can be derived by

- (a) choosing the sample (or nodal) points x_i , $0 \leq i \leq m$, equidistant on $[a, b]$, and
- (b) choosing the weights α_i , $0 \leq i \leq m$, so that numerical quadrature is exact for the highest degree polynomial possible.

(We saw that, using $(m + 1)$ points, the degree of precision for Newton–Cotes formulas is m if m is odd and $(m + 1)$ if m is even.)

In Gaussian quadrature, the points and weights x_i , α_i , $0 \leq i \leq m$, are *both* chosen so that the quadrature formula is exact for the highest degree polynomial possible. This results in the degree of precision for $(m + 1)$ -point Gaussian quadrature being $2m + 1$. Consider the following example.

Example 6.9

Take $J(f) = \int_a^b f(x)dx$ and $m = 1$. By a change of variables, we convert to the interval $[-1, 1]$ as follows:

$$J(f) = \int_a^b f(x)dx \tag{6.29}$$

$$\begin{aligned} &= \int_{-1}^1 \left(\frac{b-a}{2} \right) f \left(\frac{z(b-a) + a + b}{2} \right) dz \\ &= \int_{-1}^1 g(z)dz = J(g) \quad \left(\text{where } z = \frac{2x - a - b}{b - a} \right). \end{aligned} \tag{6.30}$$

We want to find α_0 , α_1 , z_1 , and z_2 such that $Q(g) = J(g)$ for the highest degree polynomial possible. Letting $g(z) = 1$, $g(z) = z$, $g(z) = z^2$, and

$g(z) = z^3$, we obtain the following nonlinear system:

$$\begin{aligned}\int_{-1}^1 1 \, dz &= 2 = \alpha_0 + \alpha_1 \\ \int_{-1}^1 z \, dz &= 0 = \alpha_0 z_0 + \alpha_1 z_1 \\ \int_{-1}^1 z^2 \, dz &= \frac{2}{3} = \alpha_0 z_0^2 + \alpha_1 z_1^2 \\ \int_{-1}^1 z^3 \, dz &= 0 = \alpha_0 z_0^3 + \alpha_1 z_1^3.\end{aligned}$$

Solving, we obtain $\alpha_0 = \alpha_1 = 1$, $z_0 = -1/\sqrt{3}$, $z_1 = 1/\sqrt{3}$, which are the 2-point Gaussian points and weights. Hence,

$$\int_a^b f(x)dx \approx \frac{b-a}{2} \left[\alpha_0 f\left(\frac{z_0(b-a)+a+b}{2}\right) + \alpha_1 f\left(\frac{z_1(b-a)+a+b}{2}\right) \right].$$

□

We now consider more sophisticated ways to determine these points and weights, as well as error estimates. Let

$$J(f) = \int_a^b f(x)\rho(x)dx, \quad (6.31)$$

where $\rho(x)$ is a real, positive, piecewise continuous weight function on (a, b) , e.g. $\rho(x) = \sqrt{x}$ on $(0, 1)$.

REMARK 6.8 We are generalizing the problem from $\rho(x) = 1$. □

We consider quadrature formulas of the type

$$Q(f) = (b-a) \sum_{k=0}^m \alpha_k f(x_k) \quad (6.32)$$

with sample points x_k , $0 \leq k \leq m$ and weights α_k , $0 \leq k \leq m$.

DEFINITION 6.2 We call $Q(f)$ a Gaussian quadrature formula if

$$\int_a^b x^j \rho(x)dx = (b-a) \sum_{k=0}^m \alpha_k x_k^j$$

for $j = 0, 1, 2, \dots, 2m+1$. That is, the quadrature formula has degree of precision $2m+1$, i.e., it is exact for polynomials of degree $\leq 2m+1$.

To find the points and weights, we introduce the inner product

$$(f, g) = \int_a^b \rho(x) f(x) g(x) dx.$$

DEFINITION 6.3 The set of polynomials $\{p_i(x)\}_{i=0}^\infty$ is called orthogonal on $[a, b]$ with respect to weight function $\rho(x)$ if

$$\int_a^b \rho(x) p_i(x) p_j(x) dx = 0$$

whenever $i \neq j$, i.e., if $(p_i, p_j) = 0$ for $i \neq j$.

Recall the Gram–Schmidt orthogonalization process to find the set $\{p_i\}_{i=0}^\infty$ of orthogonal polynomials: Let

$$p_0(x) = 1, \text{ and } p_j(x) = x^j - \sum_{k=0}^{j-1} \frac{1}{\|p_k\|^2} (x^j, p_k) p_k(x) \quad \text{for } j = 1, 2, \dots, \quad (6.33)$$

where $\|p_k\|^2 = (p_k, p_k)$.

REMARK 6.9 $p_j(x)$ is of degree j . □

We will see that the Gaussian quadrature points are the zeros of $p_{m+1}(x)$. First, we show that the zeros of $\{x_i\}_{i=0}^m$ of $p_{m+1}(x)$ are distinct and lie on $[a, b]$.

PROPOSITION 6.1

Let $\{p_i\}_{i=0}^\infty$ be the sequence (6.33) of orthogonal polynomials. If f is any continuous function on $[a, b]$ that is orthogonal to $p_0, p_1, \dots, p_{k-1}$, then f must either vanish identically or change sign at least k times in (a, b) .

PROOF Since $p_0(x) = 1$, $\int_a^b f(x) \rho(x) dx = 0$. Since $\rho(x) > 0$ on (a, b) , if $f(x) \neq 0$, then $f(x)$ must change sign at least once in (a, b) . Suppose that f changes sign fewer than k times, say at $r_1 < r_2 < \dots < r_s$, $s < k$. Then, on each interval (a, r_1) , (r_1, r_2) , $\dots$, (r_s, b) , f does not change sign, but has opposite signs on adjacent subintervals. Consider $p(x) = (x - r_1)(x - r_2) \dots (x - r_s)$. This polynomial p shares with f the property that it changes sign at r_i , $1 \leq i \leq s$, and has opposite signs on adjacent intervals. Hence, $\int_a^b f(x) \rho(x) p(x) dx \neq 0$. Now $p(x)$ can be written as a linear combination $p_0, p_1, \dots, p_s$, $s < k$. But $\int_a^b f(x) \rho(x) p_i(x) dx = 0$ for $i = 0, 1, \dots, k-1$. Thus, $\int_a^b f(x) \rho(x) p(x) dx = 0$, a contradiction. Therefore, f must vanish at least k times in (a, b) . □

COROLLARY 6.1

Let $\{p_i\}_{i \geq 0}$ be the sequence of orthogonal polynomials given by (6.33). The roots of $p_i(x)$ are simple and lie in (a, b) .

PROOF The polynomial p_i is orthogonal to $p_0, p_1, \dots, p_{i-1}$. Thus by, the proposition, we conclude that $p_i(x)$ must vanish at least i times in (a, b) , and hence identically i times, since p_i is of degree i . $\square$

REMARK 6.10 By Corollary 6.1, the zeros $\{x_i\}_{i=0}^m$ of $p_{m+1}(x)$ are distinct and lie in (a, b) . $\square$

Now consider $\int_a^b \rho(x)f(x)dx$. If $f(x)$ is a polynomial of degree m , we can write $f(x)$ as

$$\sum_{j=0}^m f(x_j)L_j(x), \quad \text{where} \quad L_j(x) = \prod_{\substack{i=0 \\ i \neq j}}^m \frac{x - x_i}{x_j - x_i}.$$

(This is the Lagrange form of interpolating polynomial for any $m+1$ distinct points $x_0, x_1, \dots, x_m$ on (a, b) .) Thus, $\int_a^b \rho(x)f(x)dx \approx (b-a) \sum_{j=0}^m \alpha_j f(x_j)$ will be exact for polynomials of degree $\leq m$ provided,

$$\alpha_j = \frac{1}{b-a} \int_a^b \rho(x) \prod_{\substack{i=0 \\ i \neq j}}^m \frac{x - x_i}{x_j - x_i} dx = \frac{1}{b-a} \int_a^b \rho(x)L_j(x)dx. \quad (6.34)$$

We now show that the Gaussian quadrature points $\{x_i\}_{i=0}^m$ are the zeros of $p_{m+1}(x)$ and the weights $\{\alpha_j\}_{j=0}^m$ satisfy (6.34).

THEOREM 6.1

Suppose that the $(m+1)$ -point quadrature formula is exact for polynomials of degree $\leq m$. A quadrature formula is a Gaussian quadrature formula (exact for polynomials of degree $\leq 2m+1$) if and only if the sample points $x_0, x_1, \dots, x_m$ are the zeros of the orthogonal polynomial $p_{m+1}(x)$, and the coefficients $\alpha_0, \alpha_1, \dots, \alpha_m$ can be expressed as in (6.34).

PROOF Let $\{x_i\}_{i=0}^m$ be the zeros of $p_{m+1}(x)$. Let $\varphi(x)$ be a polynomial of degree $\leq 2m+1$. Then, $\varphi(x) = q(x)p_{m+1}(x) + r(x)$ (by dividing φ by p_{m+1}), where the degrees of q and r are each $\leq m$. Therefore, $(p_{m+1}, q) = 0$, so

$$\int_a^b \rho(x)\varphi(x)dx = \int_a^b \rho(x)r(x)dx = (b-a) \sum_{i=0}^m \alpha_i r(x_i),$$

since r is of degree $\leq m$. Hence,

$$\int_a^b \rho(x)\varphi(x)dx = (b-a) \sum_{i=0}^m \alpha_i \varphi(x_i),$$

since $p_{m+1}(x_i) = 0$ for $i = 0, 1, \dots, m$. Therefore, the integration formula is exact for $\varphi(x)$.

Conversely, let $Q(f)$ be exact for $f \in P_{2m+1}$. In particular, choose $f(x) = p_j(x) \prod_{i=0}^m (x - x_i)$ for $j \leq m$. Then,

$$\int_a^b \rho(x)f(x)dx = (b-a) \sum_{i=0}^m \alpha_i f(x_i) = 0.$$

We thus conclude that for $j \leq m$,

$$\int_a^b \rho(x)p_j(x) \prod_{i=0}^m (x - x_i)dx = 0,$$

i.e.

$$\left(\prod_{i=0}^m (x - x_i), p_j(x) \right) = 0 \quad \text{for } j = 0, 1, \dots, m.$$

Thus, $\prod_{i=0}^m (x - x_i)$ is orthogonal to $p_j(x)$, $0 \leq j \leq m$. By uniqueness of the

Gram-Schmidt process, $\prod_{i=0}^m (x - x_i)$ must be a scalar multiple of $p_{m+1}(x)$.

Hence, x_i , $0 \leq i \leq m$, are the roots of $p_{m+1}(x)$. Finally, since the integration formula is exact for polynomials of degree $\leq m$, (6.34) is satisfied. $\square$

REMARK 6.11 An efficient algebraic procedure for calculating the α_j and x_j , $0 \leq j \leq m$ can be described. Notice that

$$\int_a^b \rho(x)f(x)dx = (b-a) \sum_{i=0}^m \alpha_i f(x_i)$$

for $f(x) = 1$, $f(x) = x$, $f(x) = x^2$, $\dots$, $f(x) = x^{2m+1}$, and set

$$\hat{m}_j = \frac{1}{b-a} \int_a^b \rho(x)x^j dx.$$

Then, we have

$$\begin{aligned} \alpha_0 &+ \alpha_1 &+ \dots + \alpha_m &= \hat{m}_0, \\ \alpha_0 x_0 &+ \alpha_1 x_1 &+ \dots + \alpha_m x_m &= \hat{m}_1, \\ \alpha_0 x_0^2 &+ \alpha_1 x_1^2 &+ \dots + \alpha_m x_m^2 &= \hat{m}_2, \\ &\vdots && \\ \alpha_0 x_0^{2m+1} &+ \alpha_1 x_1^{2m+1} &+ \dots + \alpha_m x_m^{2m+1} &= \hat{m}_{2m+1}. \end{aligned} \tag{6.35}$$

Define $p(x) = (x - x_0)(x - x_1) \dots (x - x_m) = \sum_{j=0}^{m+1} c_j x^j$, with $c_{m+1} = 1$. Using the first $(m + 2)$ equations of (6.35), we have

$$\begin{aligned} c_0 \quad (\alpha_0 \quad + \alpha_1 \quad + \dots + \alpha_m) &= c_0 \quad \hat{m}_0 \\ c_1 \quad (\alpha_0 x_0 \quad + \alpha_1 x_1 \quad + \dots + \alpha_m x_m) &= c_1 \quad \hat{m}_1 \\ &\vdots \\ c_{m+1} (\alpha_0 x_0^{m+1} + \alpha_1 x_1^{m+1} + \dots + \alpha_m x_m^{m+1}) &= c_{m+1} \hat{m}_{m+1} \end{aligned} \quad (6.36)$$

Hence,

$$\begin{aligned} &\hat{m}_0 c_0 + \hat{m}_1 c_1 + \dots + \hat{m}_{m+1} c_{m+1} \\ &= \alpha_0 (c_0 + c_1 x_0 + \dots + c_{m+1} x_0^{m+1}) \\ &\quad + \alpha_1 (c_0 + c_1 x_1 + \dots + c_{m+1} x_1^{m+1}) \\ &\quad \vdots \\ &\quad + \alpha_m (c_0 + c_1 x_m + \dots + c_{m+1} x_m^{m+1}) \\ &= \alpha_0 p(x_0) + \alpha_1 p(x_1) + \dots + \alpha_m p(x_m) \\ &= 0, \quad \text{since } p(x_j) = 0 \text{ for } 0 \leq j \leq m. \end{aligned}$$

Thus, $\hat{m}_0 c_0 + \hat{m}_1 c_1 + \dots + \hat{m}_m c_m = -\hat{m}_{m+1}$. Considering the second through $(m + 3)$ rd equations in (6.35) in the same manner yields:

$$\hat{m}_1 c_0 + \hat{m}_2 c_1 + \dots + \hat{m}_{m+1} c_m = -\hat{m}_{m+2}$$

Continuing in this manner, the following system is obtained:

$$\begin{aligned} \hat{m}_0 c_0 + \quad \hat{m}_1 c_1 + \dots + \quad \hat{m}_m c_m &= -\hat{m}_{m+1} \\ \hat{m}_1 c_0 + \quad \hat{m}_2 c_1 + \dots + \hat{m}_{m+1} c_m &= -\hat{m}_{m+2} \\ \hat{m}_2 c_0 + \quad \hat{m}_3 c_1 + \dots + \hat{m}_{m+2} c_m &= -\hat{m}_{m+3} \\ &\vdots \\ \hat{m}_m c_0 + \hat{m}_{m+1} c_1 + \dots + \quad \hat{m}_{2m} c_m &= -\hat{m}_{2m+1} \end{aligned} \quad (6.37)$$

Also, since $c_{m+1} = 1$, the polynomial $p(x)$ is completely determined when (6.37) is solved for $c_0, c_1, \dots, c_m$. Hence, if the roots $\{x_i\}_{i=0}^m$ of $p(x)$, which are distinct and lie on (a, b) , are calculated, then the weights α_j , $0 \leq j \leq m$, can be determined from (6.35). $\square$

6.3.3.1 Examples of Gauss-Quadrature Rules

Suppose that $\rho(x) = 1$ (standard Gauss-Legendre quadrature rules). First,

$$\begin{aligned} \int_a^b f(x) dx &= \frac{b-a}{2} \int_{-1}^1 f\left(\frac{z(b-a) + a+b}{2}\right) dz = \int_{-1}^1 g(z) dz \\ &\approx 2 \sum_{j=0}^m \alpha_j g(z_j), \end{aligned}$$

where $g(z) = \frac{b-a}{2}f((z(b-a) + a + b)/2)$. Thus, we need only consider the interval $[-1, 1]$, since if α_j and z_j , $0 \leq j \leq n$, are the weights and points for $[-1, 1]$, then, α_j and $(z_j(b-a) + a + b)/2$, $0 \leq j \leq m$, are the weights and points for $[a, b]$. The weights and points for the first few Gauss–Legendre quadrature rules, where the α_j do not include the factor of 2 in the preceding formula, appear in Table 6.1.

TABLE 6.1: Weights and sample points: Gauss–Legendre quadrature

1 point ($m = 0$)	$\alpha_1 = 1, z_1 = 0$ (midpoint rule)
2 point ($m = 1$)	$\alpha_1 = \alpha_2 = 1/2, z_1 = -\frac{1}{\sqrt{3}}, z_2 = \frac{1}{\sqrt{3}}$
3 point ($m = 2$)	$\alpha_1 = \frac{5}{18}, \alpha_2 = \frac{8}{18}, \alpha_3 = \frac{5}{18},$ $z_1 = -\sqrt{\frac{3}{5}}, z_2 = 0, z_3 = \sqrt{\frac{3}{5}}$
4 point ($m = 3$)	$\alpha_1 = \frac{1}{4} - \frac{1}{6\sqrt{4.8}}, \alpha_2 = \frac{1}{4} + \frac{1}{6\sqrt{4.8}}, \alpha_3 = \alpha_2, \alpha_4 = \alpha_1,$ $z_1 = -\sqrt{\frac{3+\sqrt{4.8}}{7}}, z_2 = -\sqrt{\frac{3-\sqrt{4.8}}{7}}, z_3 = -z_2, z_4 = -z_1.$

6.3.3.2 Error Formula for Gaussian Quadrature

To derive an error formula for Gaussian quadrature, we start with the following lemma.

LEMMA 6.1

Let $x_0, x_1, \dots, x_m$ be distinct points in $[a, b]$ and let $f \in C^{2m+2}[a, b]$. If p is the Hermite interpolating polynomial of a most degree $2m + 1$ such that

$$p(x_i) = f(x_i), \quad p'(x_i) = f'(x_i) \quad \text{for } i = 0, 1, \dots, m,$$

then

$$f(x) - p(x) = \frac{f^{(2m+2)}(\xi(x))}{(2m+2)!} \prod_{i=0}^m (x - x_i)^2 \quad \text{where } \xi(x) \in (a, b).$$

PROOF Lemma 6.1 is a restatement of Theorem 4.8, the proof of which appears on page 221. $\square$

We now have

THEOREM 6.2

Let $\rho(x) > 0$ be a weight function defined on $[a, b]$ and let $\{p_k(x)\}_{k=0}^\infty$ be

the associated sequence of orthogonal polynomials generated by (6.33). Let $\{x_i, \alpha_i\}_{i=0}^m$ be the points and weights for Gauss quadrature. Then, if $f \in C^{2m+2}[a, b]$, we have

$$\int_a^b \rho(x) f(x) dx - (b-a) \sum_{j=0}^m \alpha_j f(x_j) = \frac{f^{(2m+2)}(\xi)}{(2m+2)!} \int_a^b \rho(x) p_{m+1}^2(x) dx, \quad (6.38)$$

where $\xi \in (a, b)$.

PROOF Let $h_{2m+1}(x)$ be the Hermite interpolating polynomial of degree at most $2m+1$ that interpolates the function values and slopes of $f(x)$ at x_j , $0 \leq j \leq m$, i.e., $f(x_j) = h_{2m+1}(x_j)$, $f'(x_j) = h'_{2m+1}(x_j)$, $0 \leq j \leq m$. By lemma 6.1,

$$f(x) - h_{2m+1}(x) = \frac{f^{(2m+2)}(\xi(x))(x-x_0)^2(x-x_1)^2 \dots (x-x_m)^2}{(2m+2)!}$$

for some $\xi(x) \in (a, b)$. Multiplying both sides by $\rho(x)$ and integrating,

$$\int_a^b (f(x) - h_{2m+1}(x)) \rho(x) dx = \frac{1}{(2m+2)!} \int_a^b \rho(x) f^{(2m+2)}(\xi(x)) p_{m+1}^2(x) dx,$$

since $p_{m+1}^2(x) = (x-x_0)^2(x-x_1)^2 \dots (x-x_m)^2$ and the leading coefficient of $p_{m+1}(x)$ is 1. Thus,

$$\int_a^b f(x) \rho(x) dx - \int_a^b h_{2m+1}(x) \rho(x) dx = \frac{f^{(2m+2)}(\xi)}{(2m+2)!} \int_a^b \rho(x) p_{m+1}^2(x) dx$$

by the mean value theorem for integrals. Now,

$$\int_a^b \rho(x) h_{2m+1}(x) dx = (b-a) \sum_{j=0}^m \alpha_j h_{2m+1}(x_j) = (b-a) \sum_{j=0}^m \alpha_j f(x_j)$$

(since the formula is exact for polynomials of degree at most $2m+1$). Thus,

$$E(f) = J(f) - Q(f) = \frac{f^{(2m+2)}(\xi)}{(2m+2)!} \int_a^b \rho(x) p_{m+1}^2(x) dx.$$

□

REMARK 6.12 Consider

$$\int_a^b \rho(x) p_{m+1}^2(x) dx = \int_a^b \rho(x) \prod_{i=0}^m (x-x_i)^2 dx.$$

Let

$$z = \frac{x-a}{b-a}, \quad z_i = \frac{x_i-a}{b-a}, \quad (x-x_i) = (z-z_i)(b-a).$$

Then,

$$\int_a^b \rho(x) p_{m+1}^2(x) dx = (b-a)^{2m+3} \int_0^1 \rho(z(b-a)+a) \prod_{i=0}^m (z-z_i)^2 dz.$$

Hence,

$$E(f) = \frac{f^{(2m+2)}(\xi)}{(2m+2)!} H^{2m+3} \tilde{\beta}_m,$$

where

$$H = (b-a) \quad \text{and} \quad \tilde{\beta}_m = \int_0^1 \rho(z(b-a)+a) \prod_{i=0}^m (z-z_i)^2 dz.$$

□

REMARK 6.13 By Theorem 6.2, the degree of precision of $(m+1)$ -point Gauss quadrature is $2m+1$. For example, 4-point Gauss quadrature integrates polynomials of degree less than or equal to 7 exactly. □

REMARK 6.14 Suppose that $\rho(x) = 1$. For composite Gauss quadrature,

$$\begin{aligned} \int_a^b f(x) dx &= \sum_{\nu=0}^{N-1} \frac{h}{2} \int_{-1}^1 f\left(\frac{(z+1)h}{2} + a + h\nu\right) dz \\ &\approx h \sum_{\nu=0}^{N-1} \sum_{j=0}^m \alpha_j f\left(\frac{(z_j+1)h}{2} + a + h\nu\right), \end{aligned}$$

where $\sum_{j=0}^m \alpha_j = 1$ and α_j, z_j are Gauss quadrature weights and points for $[-1, 1]$ and $h = \frac{b-a}{N}$. Furthermore, from the derivation of composite quadrature (page 335) and Theorem 6.2, the error in composite $(m+1)$ -point Gauss quadrature is proportional to h^{2m+2} for $f \in C^{2m+2}[a, b]$. □

There is one additional aspect about Gauss quadrature which is important. That is, $\alpha_j > 0$ for $j = 0, 1, 2, \dots, m$ for any m . This implies, since

$$\int_a^b \rho(x) dx = \sum_{j=0}^m \alpha_j \quad \text{for} \quad f(x) = 1,$$

that

$$\sum_{j=0}^m |\alpha_j| = \int_a^b \rho(x) dx,$$

which is constant for all m . This makes Gauss quadrature stable in the presence of rounding errors, in contrast to Newton–Cotes methods. In addition, as you show in Exercise 12 on page 378, $Q_m(f) \rightarrow J(F)$ as $m \rightarrow \infty$ for any $f \in C[a, b]$, where $Q_m(f)$ represents m -point Gaussian quadrature with weight function ρ . Hence, a composite procedure is not necessary for convergence of Gauss quadrature for $f \in C[a, b]$. In fact, in some applications, 64-point or higher Gauss quadrature rules are used. Nonetheless, since accurate values of weights and sample points may be difficult to obtain for very high-order Gaussian quadrature, composite Gauss quadrature is useful. In contrast, as mentioned earlier, $E(f)$ may not go to zero for Newton–Cotes formulas⁶ as $m \rightarrow \infty$, even for $f \in C^\infty[a, b]$. Thus, composite quadrature is essential for Newton–Cotes rules. (Recall that, in Exercise 11 on page 377, you show that $E(f) \rightarrow 0$ as $h \rightarrow 0$ for general composite quadrature for $f \in C[a, b]$, be it Gaussian or Newton–Cotes.)

THEOREM 6.3

The weights α_j , $j = 0, 1, \dots, m$, in Gaussian quadrature are all positive for any m .

PROOF Recall that $(m+1)$ -point Gauss quadrature is exact for polynomials of degree less than or equal to $2m+1$. Let

$$q_j(x) = \frac{p_{m+1}^2(x)}{(x - x_j)^2} = \prod_{\substack{i=0 \\ i \neq j}}^m (x - x_i)^2.$$

Since $q_j \in P_{2m}$,

$$\begin{aligned} \int_a^b \rho(x) q_j(x) dx &= \sum_{i=0}^m \alpha_i q_j(x_i) (b-a) = (b-a) \alpha_j \prod_{\substack{i=0 \\ i \neq j}}^n (x_j - x_i)^2 \\ &= (b-a) \alpha_j (p'_{m+1}(x_j))^2, \end{aligned}$$

since $q_j(x_i) = 0$ if $i \neq j$. Thus,

$$\alpha_j = \frac{\int_a^b \rho(x) p_{m+1}^2(x) / (x - x_j)^2 dx}{(p'_{m+1}(x_j))^2 (b-a)} > 0$$

for each j . □

⁶which have the advantage of uniformly spaced sample points

6.3.4 Romberg Integration

Romberg integration is an efficient and popular numerical integration procedure. Let

$$J(f) = \int_a^b f(x)dx,$$

and let $S_h(f)$ be some composite numerical integration rule using interval width h . For example, using the notation of section 6.3.1 (starting on page 333),

$$S_h(f) = \sum_{\nu=0}^{N-1} Q_{\nu}(f) = h \sum_{\nu=0}^{N-1} \sum_{j=0}^m f\left(\frac{hx_j + 2a + (2\nu + 1)h}{2}\right) \alpha_j,$$

$$\text{where } h = \frac{b-a}{N} \quad \text{and } x_j \in [-1, 1] \text{ for } 0 \leq j \leq m.$$

Let $E_h(f) = J(f) - S_h(f)$. Now, suppose that

$$E_h(f) = c_1 h + c_2 h^2 + \cdots + c_{2n-2} h^{2n-2} + \mathcal{O}(h^{2n-1}), \quad (6.39)$$

where the c_i , $i = 0, 1, \dots, 2n-2$ are constants independent of h . Consider

$$\begin{aligned} J(f) &= S_h(f) + c_1 h + c_2 h^2 + \cdots + c_{2k-2} h^{2k-2} + \mathcal{O}(h^{2k-1}), \text{ so} \\ J(f) &= S_{h/2}(f) + c_1 \left(\frac{h}{2}\right) + c_2 \left(\frac{h}{2}\right)^2 + \cdots, \end{aligned}$$

where $S_{h/2}$ is the composite rule approximation using interval width $h/2$. Thus,

$$2J(f) - J(f) = 2S_{h/2}(f) - S_h(f) + c'_2 h^2 + c'_3 h^3 + \cdots$$

Set $S_h^{(1)}(f) = 2S_{h/2} - S_h(f)$. Then,

$$\begin{aligned} J(f) &= S_h^{(1)}(f) + c'_2 h^2 + c'_3 h^3 + c'_4 h^4 + \cdots, \text{ and} \\ J(f) &= S_{h/2}^{(1)}(f) + c'_2 \left(\frac{h}{2}\right)^2 + c'_3 \left(\frac{h}{2}\right)^3 + \cdots \end{aligned}$$

Thus,

$$J(f) = \frac{4S_{h/2}^{(1)}(f) - S_h^{(1)}(f)}{3} + c''_3 h^3 + c''_4 h^4 + \cdots$$

Set

$$S_h^{(2)}(f) = \left(4S_{h/2}^{(1)}(f) - S_h^{(1)}(f)\right) / 3.$$

Continuing this procedure, set

$$S_h^{(k+1)}(f) = \frac{2^{k+1} S_{h/2}^{(k)}(f) - S_h^{(k)}(f)}{2^{k+1} - 1}. \quad (6.40)$$

We can write these approximations as in Table 6.2. In this table, the quantities in a particular column are computed from the two quantities in the preceding column immediately to the left and above, using (6.40).

TABLE 6.2: Illustration of Richardson extrapolation

$\mathcal{O}(h)$	$\mathcal{O}(h^2)$	$\mathcal{O}(h^3)$	$\mathcal{O}(h^4)$
$S_h^{(0)} = S_h(f)$			
$S_{h/2}^{(0)} = S_{h/2}(f)$	$S_h^{(1)}(f)$		
$S_{h/4}^{(0)} = S_{h/4}(f)$	$S_{h/2}^{(1)}(f)$	$S_h^{(2)}(f)$	
$S_{h/8}^{(0)} = S_{h/8}(f)$	$S_{h/4}^{(1)}(f)$	$S_{h/2}^{(2)}(f)$	$S_h^{(3)}(f)$

If the error expansion is of even order, which is preferable, i.e.,

$$E_h(f) = c_2h^2 + c_4h^4 + \cdots + c_{2k-2}h^{2k-2} + \mathcal{O}(h^{2k-1}), \tag{6.41}$$

then

$$\begin{aligned} S_h^{(1)}(f) &= (4S_{h/2} - S_h)/3 \\ S_h^{(2)}(f) &= (16S_{h/2}^{(1)} - S_h^{(1)})/15 \\ &\vdots \end{aligned}$$

which, in general, is

$$S_h^{(k+1)}(f) = \frac{4^{k+1}S_{h/2}^{(k)}(f) - S_h^{(k)}(f)}{4^{k+1} - 1}. \tag{6.42}$$

The computations associated with (6.42) appear in Table 6.3.

TABLE 6.3: Illustration of even-order Richardson extrapolation

$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$
$S_h^{(0)} = S_h(f)$			
$S_{h/2}^{(0)} = S_{h/2}(f)$	$S_h^{(1)}(f)$		
$S_{h/4}^{(0)} = S_{h/4}(f)$	$S_{h/2}^{(1)}(f)$	$S_h^{(2)}(f)$	
$S_{h/8}^{(0)} = S_{h/8}(f)$	$S_{h/4}^{(1)}(f)$	$S_{h/2}^{(2)}(f)$	$S_h^{(3)}(f)$

REMARK 6.15 The above process is called *Richardson extrapolation*, and can be performed whenever the error expansion has the form (6.39) or

(6.41). Richardson extrapolation is useful in numerical integration, numerical solution of integral equations and initial-value problems, numerical methods for solving partial differential equations, and stochastic differential equations. $\square$

REMARK 6.16 When Richardson extrapolation is applied to the composite trapezoidal rule, the numerical integration procedure is called *Romberg integration*. (However, as noted later, this extrapolation process can be applied to all standard composite integration formulas.) $\square$

We now consider the composite trapezoidal rule in detail, i.e.,

$$\begin{aligned} S_h(f) &= h \left(\frac{1}{2}f(a) + \frac{1}{2}f(b) \right) + h \sum_{\nu=1}^{N-1} f(a + \nu h) \\ &= \frac{h}{2} \sum_{\nu=0}^{N-1} [f(a + \nu h) + f(a + \nu h + h)], \quad \text{where } h = \frac{b-a}{N}. \end{aligned} \quad (6.43)$$

We need to show that

$$E_h(f) = c_2 h^2 + c_4 h^4 + c_6 h^6 + \dots,$$

where the constants c_i , $i = 2, 4, \dots$, do not depend on h . To establish this result, we will use the following lemma.

LEMMA 6.2

(Euler–Maclaurin formula) For $f \in C^{2m+2}[a, b]$,

$$\begin{aligned} \int_a^b f(x) dx &= \frac{b-a}{2} [f(a) + f(b)] - \sum_{k=0}^m \frac{B_{2k}}{(2k)!} H^{2k} [f^{(2k-1)}(b) - f^{(2k-1)}(a)] \\ &\quad - \frac{H^{2m+3} B_{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi), \end{aligned}$$

where $H = b - a$ and ξ is some point in $[a, b]$. The numbers $B_2, B_4, B_6, \dots$ are the Bernoulli numbers of even order, which can be defined by

$$\coth x = \frac{1}{x} \left\{ B_0 + \sum_{k=1}^{\infty} B_{2k} \frac{(2x)^{2k}}{(2k)!} \right\} = \frac{1}{x} \left\{ 1 + \frac{1}{3}x^2 - \frac{1}{45}x^4 + \dots \right\},$$

i.e.,

$$B_0 = 1, \quad B_2 = \frac{1}{6}, \quad B_4 = -\frac{1}{30}, \quad B_6 = \frac{1}{42}, \quad B_8 = -\frac{1}{30}, \quad \dots$$

PROOF Let $P_n(t)$, $n = 0, 1, 2, \dots$ be the Bernoulli polynomials defined by $P_0(t) = 1$ and

$$\sum_{k=0}^n \binom{n+1}{k} P_k(t) = (n+1)t^n, \quad (6.44)$$

i.e.,

$$(n+1)P_n(t) + \sum_{k=0}^{n-1} \binom{n+1}{k} P_k(t) = (n+1)t^n.$$

For example, the first four of these are

$$\begin{aligned} P_0(t) &= 1, \\ P_1(t) &= t - \frac{1}{2}, \\ P_2(t) &= t^2 - t + \frac{1}{6}, \\ P_3(t) &= t^3 - \frac{3}{2}t^2 + \frac{1}{2}t. \end{aligned}$$

We will establish first the following properties of Bernoulli polynomials:

$$P'_n(t) = nP_{n-1}(t), \quad n \geq 1. \quad (6.45)$$

$$P_n(t+1) - P_n(t) = nt^{n-1}. \quad (6.46)$$

$$P_n(t) = \sum_{k=0}^n \binom{n}{k} P_k(0)t^{n-k}. \quad (6.47)$$

$$P_n(1-t) = (-1)^n P_n(t). \quad (6.48)$$

Proof of (6.45): We establish (6.45) by induction. Clearly (6.45) is valid for $n = 1$. Now, suppose $P'_k(t) = kP_{k-1}(t)$ for $k = 1, 2, \dots, n-1$. Differentiating (6.44) yields

$$\sum_{k=0}^n \binom{n+1}{k} P'_k(t) = n(n+1)t^{n-1} = (n+1) \sum_{k=0}^{n-1} \binom{n}{k} P_k(t). \quad (6.49)$$

By the induction hypothesis,

$$\sum_{k=1}^{n-1} \binom{n+1}{k} kP_{k-1}(t) + (n+1)P'_n(t) = (n+1) \sum_{k=0}^{n-1} \binom{n}{k} P_k(t). \quad (6.50)$$

Thus,

$$P'_n(t) = \sum_{k=0}^{n-1} \binom{n}{k} P_k(t) - \sum_{k=1}^{n-1} \binom{n}{k-1} P_{k-1}(t),$$

since

$$\frac{k}{n+1} \binom{n+1}{k} = \binom{n}{k-1}.$$

Hence,

$$P'_n(t) = nP_{n-1}(t) + \sum_{k=0}^{n-2} \binom{n}{k} P_k(t) - \sum_{k=1}^{n-1} \binom{n}{k-1} P_{k-1}(t) = nP_{n-1}(t).$$

Proof of (6.46): By (6.45),

$$\begin{aligned} P''_n(t) &= nP'_{n-1}(t) = n(n-1)P_{n-2}(t), \\ P'''_n(t) &= n(n-1)(n-2)P_{n-3}(t), \\ &\vdots \\ P_n^{(k)}(t) &= n(n-1)\dots(n-k+1)P_{n-k}(t). \end{aligned}$$

Thus, the Taylor series for $P_n(t)$ is

$$P_n(t+h) = \sum_{k=0}^n \frac{1}{k!} P_n^{(k)}(t) h^k = \sum_{k=0}^n \binom{n}{k} P_{n-k}(t) h^k. \quad (6.51)$$

Now, set $h = 1$ and use $\binom{n}{k} = \binom{n}{n-k}$ and (6.44) to obtain

$$\begin{aligned} P_n(t+1) &= \sum_{k=0}^n \binom{n}{k} P_{n-k}(t) \\ &= \sum_{k=0}^n \binom{n}{n-k} P_{n-k}(t) \\ &= P_n(t) + \sum_{k=1}^n \binom{n}{n-k} P_{n-k}(t) \\ &= P_n(t) + \sum_{j=n-1}^0 \binom{n}{j} P_j(t) \quad (\text{letting } j = n-k) \\ &= P_n(t) + n t^{n-1}. \end{aligned}$$

Hence, (6.46) is established.

Proof of (6.47): Set $h = t$ and $t = 0$ in (6.51) to obtain

$$\begin{aligned} P_n(t) &= \sum_{k=0}^n \binom{n}{k} P_{n-k}(0) t^k \\ &= \sum_{j=0}^n \binom{n}{n-j} P_j(0) t^{n-j} \quad (\text{setting } j = n-k) \\ &= \sum_{j=0}^n \binom{n}{j} P_j(0) t^{n-j}. \end{aligned}$$

Thus, $P_n(t) = \sum_{k=0}^n \binom{n}{k} P_k(0)t^{n-k}$.

Proof of (6.48): (6.46) with t replaced with $(-t)$ gives

$$P_n(-t+1) - P_n(-t) = n(-1)^{n-1}t^{n-1} = (-1)^{n-1}(P_n(t+1) - P_n(t)).$$

Writing this as

$$(-1)^n P_n(t+1) - P_n(-t) = (-1)^n P_n(t) - P_n(1-t) = F(t),$$

we see that $F(t+1) = F(t)$ for all t . Thus, F is periodic with period 1. But F is a polynomial, so F must be constant. Therefore,

$$-P_n(1-t) + (-1)^n P_n(t) = c_n.$$

Differentiating this expression and using (6.45) gives

$$(-1)^n P'_n(t) + P'_n(1-t) = (-1)^n n P_{n-1}(t) + n P_{n-1}(1-t) = 0.$$

Thus, $P_n(1-t) = (-1)^n P_n(t)$.

REMARK 6.17 Setting $t = 0$ in (6.46) and (6.48) gives

$$P_n(1) = P_n(0) = (-1)^n P_n(0)$$

for $n > 1$. Thus, $0 = P_3(0) = P_5(0) = P_7(0) = \dots$ □

REMARK 6.18 $P_n(0) = P_n(1) = B_n$ (the n -th Bernoulli number) Also, for n odd, $B_n = 0$. □

We now complete the proof of Lemma 6.2. Consider

$$\int_a^b f(x)dx = \int_0^1 g(z)dz, \quad \text{where} \quad g(z) = (b-a)f(a+(b-a)z).$$

We will show that

$$\begin{aligned} \int_0^1 g(z)dz &= \frac{1}{2}[g(0) + g(1)] - \sum_{k=0}^m \frac{B_{2k}}{(2k)!} [g(1)^{(2k-1)} - g(0)^{(2k-1)}] \\ &\quad - \frac{B_{2m+2}}{(2m+2)!} g^{(2m+2)}(\xi) \end{aligned}$$

for some $0 \leq \xi \leq 1$; it is easy to see that Lemma 6.2 follows immediately from this. Consider

$$\begin{aligned} \int_0^1 g(z)dx &= \int_0^1 g(z)P_0(z)dz = P_1(z)g(z) \Big|_{z=0}^1 - \int_0^1 P_1(z)g'(z)dz \\ &= \frac{1}{2}[g(0) + g(1)] - \int_0^1 P_1(z)g'(z)dz, \end{aligned}$$

since $P_n(z) = P'_{n+1}(z)/(n+1)$ and $P_1(1) = P_1(0) = 1/2$. Performing another integration by parts yields

$$\int_0^1 g(z)dz = \frac{1}{2}[g(0) + g(1)] - \frac{B_2}{2}[g'(1) - g'(0)] + \frac{1}{2} \int_0^1 P_2(z)g''(z)dz.$$

Continuing this procedure and using $B_{2m+1} = 0$ for $n = 0, 1, 2, \dots$,

$$\begin{aligned} \int_0^1 g(z)dz &= \frac{1}{2}[g(0) + g(1)] - \sum_{k=1}^{m+1} \frac{B_{2k}}{(2k)!} [g^{(2k-1)}(1) - g^{(2k-1)}(0)] \\ &\quad + \frac{1}{(2m+2)!} \int_0^1 P_{2m+2}(z)g^{(2m+2)}(z)dz. \end{aligned}$$

The last term in the above sum can be expressed as

$$\frac{B_{2m+2}}{(2m+2)!} [g^{(2m+1)}(1) - g^{(2m+1)}(0)] = \frac{B_{2m+2}}{(2m+2)!} \int_0^1 g^{(2m+2)}(z)dz.$$

Thus, the last term in the sum and the integral give

$$\begin{aligned} &\frac{1}{(2m+2)!} \int_0^1 (P_{2m+2}(z) - B_{2m+2})g^{(2m+2)}(z)dz \\ &= \frac{g^{(2m+2)}(\xi)}{(2m+2)!} \int_0^1 (P_{2m+2}(z) - B_{2m+2})dz \\ &= \frac{g^{(2m+2)}(\xi)}{(2m+2)!} \int_0^1 \left(\frac{P'_{2m+3}(z)}{2m+3} - B_{2m+2} \right) dz = \frac{-g^{(2m+2)}(\xi)B_{2m+2}}{(2m+2)!}, \end{aligned} \tag{6.52}$$

since $P_{2m+3}(1) = P_{2m+3}(0) = 0$.

In (6.52), we used the fact that $G(t) = P_{2n}(t) - P_{2n}(0) = P_{2n}(t) - B_{2n}$ does not change sign on $[0, 1]$. To see this, suppose that G has a zero in $(0, 1)$. Then, since $G(0) = G(1) = 0$, Rolle's theorem implies that G' has two zeros in $(0, 1)$. Since P_{2n-1} is a multiple of G' , P_{2n-1} must also have two zeros in $(0, 1)$. But P_{2n-1} also vanishes at 0 and 1, so P'_{2n-1} has three zeros in $(0, 1)$. Continuing in this manner, P_k , for odd $k < 2n$, must have at least two zeros in $(0, 1)$ as well as zeros at 0 and 1. This is impossible, since P_3 is a cubic polynomial. $\square$

Lemma 6.2 immediately gives us

THEOREM 6.4

For $f \in C^{2m+2}[a, b]$,

$$\begin{aligned} \int_a^b f(x)dx &= \frac{h}{2} \sum_{\nu=0}^{N-1} [f(x_\nu) + f(x_{\nu+1})] - \sum_{j=1}^m \frac{B_{2j}h^{2j}}{(2j)!} [f^{(2j-1)}(b) - f^{(2j-1)}(a)] \\ &\quad - \frac{(b-a)B_{2m+2}h^{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi) \end{aligned}$$

for some $\xi \in [a, b]$, where $x_\nu = a + \nu h$, $\nu = 0, 1, \dots, N$ and $h = (b - a)/N$.

PROOF Let $a = x_\nu$ and $b = x_{\nu+1}$ in Lemma 6.2. Then

$$\int_{x_\nu}^{x_{\nu+1}} f(x) dx = \frac{x_{\nu+1} - x_\nu}{2} [f(x_\nu) + f(x_{\nu+1})] - \sum_{j=1}^m \frac{B_{2j} h^{2j}}{(2j)!} [f^{(2j-1)}(x_{\nu+1}) - f^{(2j-1)}(x_\nu)] - \frac{B_{2m+2} h^{2m+3}}{(2m+2)!} f^{(2m+2)}(\xi_\nu),$$

where $h = x_{\nu+1} - x_\nu = (b - a)/N$ and $x_\nu \leq \xi_\nu \leq x_{\nu+1}$. Now sum both sides from $\nu = 0$ to $\nu = N - 1$. Then

$$\begin{aligned} \int_a^b f(x) dx &= \frac{h}{2} \sum_{\nu=0}^{N-1} [f(x_\nu) + f(x_{\nu+1})] - \sum_{j=1}^m \frac{B_{2j}}{(2j)!} h^{2j} [f^{(2j-1)}(b) - f^{(2j-1)}(a)] \\ &\quad - \frac{B_{2m+2}(Nh)h^{2m+2}}{(2m+2)!} \sum_{\nu=0}^{N-1} \frac{f^{(2m+2)}(\xi_\nu)}{N} \\ &= \frac{h}{2} \sum_{\nu=0}^N [f(x_\nu) + f(x_{\nu+1})] - \sum_{j=1}^m \frac{B_{2j}}{(2j)!} h^{2j} [f^{(2j-1)}(b) - f^{(2j-1)}(a)] \\ &\quad - \frac{B_{2m+2}(b-a)h^{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi), \end{aligned}$$

since

$$\min_{a \leq x \leq b} f^{(2m+2)}(x) \leq \sum_{\nu=0}^{N-1} \frac{f^{(2m+2)}(\xi_\nu)}{N} \leq \max_{a \leq x \leq b} f^{(2m+2)}(x).$$

Finally, this and the Intermediate Value Theorem give

$$\sum_{\nu=0}^{N-1} \frac{f^{(2m+2)}(\xi_\nu)}{N} = f^{(2m+2)}(\xi).$$

□

Now, let us examine the scheme illustrated in Table 6.2 when we use the trapezoidal rule as the base method. For this Romberg integration, define

$$S_j^{(0)}(f) = h_j \left(\frac{1}{2} f(a) + \sum_{\nu=1}^{N_j-1} f(a + \nu h_j) + \frac{1}{2} f(b) \right),$$

where

$$h_j = \frac{h}{2^j}, \quad N_j = 2^j N \quad \text{and} \quad h = \frac{b-a}{N},$$

and, in general, define

$$S_j^{(k+1)}(f) = \frac{4^{k+1}S_{j+1}^{(k)}(f) - S_j^{(k)}(f)}{4^{k+1} - 1} \quad \text{for } j = 0, 1, 2, \dots \text{ and } k = 0, 1, 2, \dots$$

Table 6.2 then becomes Table 6.4. In Table 6.4, the last row indicates that $S_j^{(1)}$

TABLE 6.4: Romberg integration and composite Newton–Cotes methods

$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$
$S_0^{(0)}$	—	—	—
$S_1^{(0)}$	$S_0^{(1)}$	—	—
$S_2^{(0)}$	$S_1^{(1)}$	$S_0^{(2)}$	—
$S_3^{(0)}$	$S_2^{(1)}$	$S_1^{(2)}$	$S_0^{(3)}$
$S_4^{(0)}$	$S_3^{(1)}$	$S_2^{(2)}$	$S_1^{(3)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
composite trapezoidal	composite Simpson's	composite closed Newton–Cotes of order $\mathcal{O}(h^6)$	$\mapsto \dots$ not Newton–Cotes methods

happens to be the composite Simpson's rule with j subintervals, while $S_j^{(2)}$ happens to be the composite closed Newton–Cotes formula of order $\mathcal{O}(h^6)$, while $S_j^{(k)}$ for $k > 2$ does not correspond to any Newton–Cotes method.

The Romberg procedure is generally continued until $|S_0^{(j-1)} - S_0^{(j)}| < \epsilon$, for a given tolerance ϵ . Then, the best estimate of the integral is $S_0^{(j)}$.

REMARK 6.19 It can be proved that if $f \in C[a, b]$, the values in each column of Table 6.4 converge to the integral [49]. $\square$

REMARK 6.20 An efficient formula for the computation is the recursion relation

$$S_{j+1}^{(0)}(f) = \frac{1}{2}(S_j^{(0)}(f) + T_j(f)),$$

where $T_j(f) = h_j \sum_{\nu=0}^{N_j-1} f(c_j + \nu h_j)$, with $c_j = a + h_j/2$. Use of this relation

requires only N_j additional computations of the function $f(x)$ to calculate $S_{j+1}^{(0)}(f)$ instead of $2N_j$. □

Example 6.10

Numerical evaluation of the sine integral. Compute

$$\int_0^1 \frac{\sin x}{x} dx.$$

Solution: With Romberg integration, we obtain the following, corresponding to Table 6.4.

1 interval	$S_0^{(0)} = 0.920735$			
2 intervals	0.939793	$S_0^{(1)} = 0.946146$		
4 intervals	0.944514	0.946088	$S_0^{(2)} = 0.946084$	
8 intervals	0.945691	0.946083	0.946083	$S_0^{(3)} = 0.946083$

Since $S_0^{(2)}$ and $S_0^{(3)}$ differ by one unit in the sixth digit, this is evidence that $S_0^{(3)}$ is accurate to six digits. □

We now consider the question: Can Richardson extrapolation be applied to other composite numerical integral rules besides the composite trapezoidal rule? It turns out that, indeed, it can be applied to basically any composite rule that integrates constant functions exactly. Before addressing this question in detail, consider the following example.

Example 6.11

The composite midpoint rule applied to the sine integral. As in Example 6.10, our task is to compute

$$\int_0^1 \frac{\sin x}{x} dx,$$

but we now use the composite midpoint rule for $S_j^{(0)}$ instead of the trapezoidal rule. We obtain

1 interval	$S_0^{(0)} = 0.958851$		
2 intervals	0.9492337	$S_0^{(1)} = 0.946028$	
4 intervals	0.9468682	0.946080	$S_0^{(2)} = 0.946083$



In [10], Baker and Hodgson show that almost any composite rule has the correct form for the error for performing Richardson extrapolation. They obtained the following result.

THEOREM 6.5

Let

$$J(f) = \int_0^1 f(x)dx, \quad Q(f) = \sum_{j=0}^n \alpha_j f(t_j), \quad \text{and}$$

$$Q_c(f) = \frac{1}{m} \sum_{i=0}^{m-1} \sum_{j=0}^n \alpha_j f((i + t_j)/m).$$

(The above is the composite rule for m intervals on $[0, 1]$, i.e., $h = 1/m$.) Suppose that $\sum_{j=0}^n \alpha_j = 1$ and $f \in C^N[0, 1]$. Then,

$$E_c(f) = J(f) - Q_c(f) = \sum_{k=1}^{N-1} c_k h^k + \mathcal{O}(h^N),$$

where the c_k 's are independent of h . (Thus, the error has the correct form for applying Richardson extrapolation.) Also,

- (a) If $Q(f) = J(f)$ for polynomials of degree $\leq r$, then $c_k = 0$ for $1 \leq k \leq r$.
- (b) If $Q(f)$ is a symmetric quadrature rule⁷ then $c_{2k+1} = 0$ for $k = 0, 1, 2, \dots$, and thus odd powers of h in the error expansion vanish.

Example 6.12

Examples of symmetric and nonsymmetric quadrature rules

- (1) The midpoint rule: $n = 0$, $\alpha_0 = 1$, $t_0 = 1/2$. This a symmetric rule with degree of precision 1. Thus, by Theorem 6.5, part (a), $c_1 = 0$, and by part (b), $c_{2k+1} = 0$ for $k = 1, 2, \dots$. Hence,

$$E_c(f) = c_2 h^2 + c_4 h^4 + c_6 h^6 + \dots,$$

the same as for the trapezoidal rule.

⁷A symmetric quadrature rule is a quadrature rule in which the points and weights are symmetric on $[0, 1]$, i.e., $\alpha_i = \alpha_{n-i}$, $t_i = 1 - t_{n-i}$ for $i = 0, 1, \dots, n$.

- (2) *The rectangle rule*: $n = 0$, $\alpha_0 = 1$, $t_0 = 1$, that is,

$$\int_0^1 f(x)dx \approx 1f(0).$$

This is not a symmetric quadrature rule, and the degree of precision is 0. Hence,

$$E_c(f) = c_1h + c_2h^2 + c_3h^3 + \dots$$

- (3) *4-point Gauss quadrature*: This is a symmetric rule. (See Table 6.1 on page 349 for the points and weights.) Also, this rule is exact for polynomials of degree ≤ 7 . Hence,

$$E_c(f) = c_8h^8 + c_{10}h^{10} + c_{12}h^{12} + \dots$$

□

6.3.5 Multiple Integrals, Singular Integrals, and Infinite Intervals

We describe some special considerations in numerical integration in this section.

6.3.5.1 Multiple Integrals

Consider

$$\int_a^b \int_c^d f(x, y) dy dx \quad \text{or} \quad \int_a^b \int_c^d \int_r^s f(x, y, z) dx dy dz.$$

How can we approximate these integrals? One way is with a *product formula*, in which we apply a one-dimensional quadrature rule in each variable. Consider

$$\begin{aligned} & \int_a^b \left(\int_c^d f(x, y) dy \right) dx \\ &= \int_a^b \sum_{\ell=0}^{L-1} \left(\int_{c_\ell}^{c_{\ell+1}} f(x, y) dy \right) dx \\ &= \int_a^b \sum_{\ell=0}^{L-1} \int_{-1}^1 f \left(x, \frac{y' h_y + 2c + (2\ell + 1)h_y}{2} \right) \frac{h_y}{2} dy' dx \\ &\approx \int_a^b \sum_{\ell=0}^{L-1} \sum_{i=0}^n \alpha_i h_y f \left(x, \frac{y_i h_y + 2c + (2\ell + 1)h_y}{2} \right) dx, \end{aligned}$$

where

$$h_y = \frac{d - c}{L}, \quad c_\ell = c + \ell h_y \quad \text{and} \quad y_i = z_i \quad \text{for } i = 0, 1, \dots, n,$$

and where the one-dimensional quadrature rule is assumed to be of the form

$$\int_{-1}^1 g(z) dz \approx 2 \sum_{i=0}^n \alpha_i g(z_i).$$

(Note that we have applied a one-dimensional composite rule over the y variable.) The integral over the x -variable is treated similarly, giving

$$\begin{aligned} & \int_a^b \int_c^d f(x, y) dy dx \\ & \approx \sum_{\ell=0}^{L-1} \sum_{k=0}^{K-1} \sum_{i=0}^n \sum_{j=0}^n \alpha_i \alpha_j h_y h_x f \left(\frac{x_j h_x + 2a + (2k+1)h_x}{2}, \frac{y_i h_y + 2c + (2\ell+1)h_y}{2} \right), \end{aligned}$$

where $h_x = (b-a)/K$, $h_y = (d-c)/L$, and $x_j = z_j$ for $j = 0, \dots, n$.

The same procedure can be used for triple or higher multiple integrals.

6.3.5.2 Singularities and Infinite Intervals

Consider $\int_a^v f(x) dx$. Suppose that f is Riemann integrable but has a singularity somewhere on $[a, b]$. (Alternately, for example, f may be continuous but f' may have a singularity on $[a, b]$, which results in low accuracy of the numerical quadrature methods used unless a large number of intervals is taken.) Without loss of generality and to fix ideas, suppose that f is Riemann integrable on $[0, 1]$, continuous on $(0, 1]$ but not continuous on $[0, 1]$. For example,

$$\int_0^1 f(x) dx = \int_0^1 \frac{x^{-1/2}}{1+x^2} dx \quad \text{or} \quad \int_0^1 f(x) dx = \int_0^1 (\cos x - 1)^{-1/3} dx.$$

There is a variety of procedures that can be used to perform the integrations numerically or to increase the accuracy of the numerical quadrature method. In particular, we can

- (i) truncate the integral,
- (ii) change variables,
- (iii) subtract out the singularity,
- (iv) ignore the singularity,
- (v) use a singular integration rule.

We now give some details for these five techniques.

(i) truncate the integral —

Procedure: Evaluate $\int_\epsilon^1 f(x) dx$ numerically and estimate $\int_0^\epsilon f(x) dx$.

Example: Suppose that

$$\int_0^1 f(x) dx = \int_0^1 \frac{g(x)}{\sqrt{x}} dx, \quad g \in C[0, 1].$$

Then,

$$\int_0^1 f(x)dx = \int_\epsilon^1 f(x)dx + \int_0^\epsilon \frac{g(x)}{\sqrt{x}}dx.$$

Hence,

$$\left| \int_0^1 f(x)dx - \int_\epsilon^1 f(x)dx \right| = \left| \int_0^\epsilon \frac{g(x)}{\sqrt{x}}dx \right| \leq \max_{0 \leq x \leq \epsilon} |g(x)| 2\sqrt{\epsilon}.$$

If $\max_{0 \leq x \leq \epsilon} |g(x)| 2\sqrt{\epsilon}$ is sufficiently small, then estimating $\int_0^1 f(x)dx$ by $\int_\epsilon^1 f(x)dx$ is reasonable.

(ii) Change variables — (may eliminate the singularity)

Example: Suppose that $f(x) = x^{-1/n}g(x)$, $n \geq 2$ and $g \in C[0, 1]$.

Let $x = t^n$, $n \geq 2$, $dx = n t^{n-1}dt$.

Then $\int_0^1 f(x)dx = \int_0^1 x^{-1/n}g(x)dx = n \int_0^1 t^{n-2}g(t^n)dt$ which is a new integral without the singularity at $t = 0$.

(iii) Subtract out the singularity —

Example 1:

$$\int_0^1 \frac{\cos x}{\sqrt{x}}dx = \int_0^1 \frac{dx}{\sqrt{x}} + \int_0^1 \frac{\cos x - 1}{\sqrt{x}}dx = 2 + \int_0^1 \frac{\cos x - 1}{\sqrt{x}}dx,$$

where the last integral is not singular at $x = 0$. (Recall $\cos x = 1 - \frac{x^2}{2} + \dots$)

Example 2:

$$\begin{aligned} & \int_0^1 \frac{x^{-1/2}}{1+x}dx \\ &= \int_0^1 \frac{x^{-1/2} + x^{1/2} - x^{1/2}}{1+x}dx \\ &= \int_0^1 \frac{x^{-1/2}(1+x)}{1+x}dx - \int_0^1 \frac{x^{1/2}}{1+x}dx = 2 - \int_0^1 \frac{x^{1/2}}{1+x}dx \\ &= 2 - \int_0^1 \frac{x^{1/2}(1+x) - x^{3/2}}{(1+x)}dx \\ &= 2 - \int_0^1 x^{1/2}dx + \int_0^1 \frac{x^{3/2}}{1+x}dx \\ &= 2 - \frac{2}{3} + \int_0^1 \frac{x^{3/2}}{1+x}dx \\ &= 2 - \frac{2}{3} + \int_0^1 \frac{x^{3/2}(1+x) - x^{5/2}}{(1+x)}dx \\ &= 2 - \frac{2}{3} + \frac{2}{5} - \int_0^1 \frac{x^{5/2}}{1+x}dx. \end{aligned}$$

(Notice that the integrand of the last integral is smooth having two continuous derivatives on $[0, 1]$). Applying the composite midpoint rule with N intervals to the above integrals yields the results in Table 6.5.

TABLE 6.5: Subtracting out the singularity, Example 2

N	$\int_0^1 \frac{x^{-1/2}}{1+x} dx$	$2 - \int_0^1 \frac{x^{1/2}}{1+x} dx$	$2 - \frac{2}{3} + \int_0^1 \frac{x^{3/2}}{1+x} dx$	$2 - \frac{2}{3} + \frac{2}{5} - \int_0^1 \frac{x^{5/2}}{1+x} dx$
1	0.94281	1.52860	1.56904	1.61548
2	1.12991	1.55256	1.56891	1.58165
4	1.26259	1.56374	1.57005	1.57345
8	1.35466	1.56820	1.57057	1.57145
16	1.41872	1.56986	1.57073	1.57096
32	1.46355	1.57046	1.57077	1.57084
64	1.49507	1.57068	1.57079	1.57081
128	1.51729	1.57075	1.57080	1.57080

- (iv) **Ignore the singularity** — In this procedure, we use a rule, such as composite Gaussian quadrature, that does not involve evaluation of $f(0)$.
Example:

$$\int_0^1 \frac{x^{-1/2}}{1+x^2} dx$$

The method works slowly. Many intervals are needed to obtain an accurate result.

- (v) **Develop singular integration rules** —

Example: Suppose that $k(x)$ is a weight function with singularity at $x = 0$ and $k(x) > 0$ for $0 < x \leq 1$. Suppose also that $\int_0^1 k(x)x^j dx$ exists for $j = 0, 1, 2, \dots, n$. We can then derive quadrature rules of interpolatory type or Gaussian rules. Given a subdivision $0 \leq x_0 < x_1 < \dots < x_n \leq 1$, we can find α_i such that

$$\int_0^1 f(x) dx = \int_0^1 k(x)g(x) dx \approx \sum_{i=0}^n \alpha_i g(x_i)$$

is exact for $g(x) = 1, x, \dots, x^n$. For example,

$$\int_0^1 f(x) dx = \int_0^1 x^{-1/2} g(x) dx \approx \frac{14}{5} g(1/3) - \frac{8}{5} g(2/3) + \frac{4}{5} g(1)$$

is exact for $g(x) = 1, g(x) = x, g(x) = x^2$.

Similarly, several procedures can be applied to numerically approximate integrals over infinite integrals:

- (i) Change variables.
- (ii) Truncate the integral.
- (iii) Use a quadrature rule appropriate for infinite intervals.

(i) Change variables —

Example:

- (a) Setting $x = e^{-y}$, $y = -\ln x$,

$$\int_0^\infty f(y)dy = \int_0^1 \frac{f(-\ln x)}{x} dx = \int_0^1 \frac{g(x)}{x} dx.$$

- (b) Setting $y = (x-a)/(b-x)$, $x = (a+by)/(1+y)$,
 $dy = (b-a)/(b-x)^2$,

$$\int_0^\infty f(y)dy = (b-a) \int_a^b \frac{f\left(\frac{x-a}{b-x}\right)}{(b-x)^2} dx.$$

- (ii) Truncate the integral —** Consider $\int_0^\infty f(x)dx \approx \int_0^R f(x)dx$, and estimate $\int_R^\infty f(x)dx$ by some other means.

Example:

$$\begin{aligned} & \left| \int_0^\infty \frac{xe^{-x^2}}{1+x^4} dx - \int_0^{10} \frac{xe^{-x^2}}{1+x^4} dx \right| \\ &= \int_{10}^\infty \frac{xe^{-x^2}}{1+x^4} dx \\ &\leq \frac{1}{1+10^4} \int_{10}^\infty xe^{-x^2} dx \\ &= \frac{1}{1+10^4} \frac{1}{2} e^{-100}, \text{ which is very small.} \end{aligned}$$

- (iii) Use a quadrature rule appropriate for an infinite interval —**
 Consider a formula of Gauss type:

$$\int_0^\infty w(x)f(x)dx \approx \sum_{k=1}^n w_k f(x_k),$$

where the points and weights have been chosen so that the approximation is exact for polynomials of degree $\leq 2n-1$. It is assumed here that $\int_0^\infty w(x)x^k dx < \infty$ for $k = 0, 1, \dots, 2n-1$. Two such examples are:

(a) $\int_0^\infty e^{-x} f(x) dx \approx \sum_{k=1}^n w_k f(x_k),$

where the points x_k are the zeros of the Laguerre polynomial $L_n(x)$ and w_k satisfy $w_k = (n!)^2 x_k / (L_{n+1}(x_k))^2$

$$(b) \int_{-\infty}^{\infty} e^{-x^2} f(x) dx \approx \sum_{k=1}^n w_k f(x_k),$$

where the points x_k are the zeros of the Hermite polynomial $H_n(x)$ and w_k satisfy $w_k = 2^{n+1} n! \sqrt{\pi} / (H_{n+1}(x_k))^2$.

See [22] for more details on numerical integration of improper integrals.

6.3.6 Monte Carlo and Quasi-Monte Carlo Methods

In multiple numerical integration, the number of computations is proportional to N^M where N is the number of intervals used in the composite quadrature rule and M is the number of iterated integrals, assuming N intervals are used for each integral. Thus, if for example $M = 6$ and $N = 100$, the number of computations is proportional to 10^{12} . For high-dimensional problems, a Monte Carlo approach can be useful. We consider here a one-dimensional problem, but the Monte Carlo approach is independent of dimension.

Let

$$J(f) = \int_0^1 f(x) dx = \int_0^1 \mu(x) f(x) dx, \quad (6.53)$$

where $\mu(x)$ is the uniform probability density on $[0, 1]$, that is, $\text{prob}(c \leq x \leq d) = \int_c^d \mu(x) dx = d - c$, assuming that $0 \leq c \leq d \leq 1$. (We can always convert $\int_a^b f(\tilde{x}) d\tilde{x}$ to $\int_0^1 f(x) dx$.) The mean value of $f(x)$ over interval $[0, 1]$ is thus $J(f)$. Let $x_1, x_2, x_3, \dots, x_n$ be n points randomly selected from a uniform distribution on $[0, 1]$. (Generally, a pseudorandom number generator is used.) We can then form the average

$$f_n = \frac{1}{n} \sum_{i=1}^n f(x_i), \quad (6.54)$$

and we would expect $f_n \rightarrow J(f)$ as $n \rightarrow \infty$. We can estimate the error in approximation (6.54) using the Central Limit Theorem.

First, the mean value of f on $[0, 1]$ is

$$J(f) = \int_0^1 f(x) \mu(x) dx = \int_0^1 f(x) dx, \quad (6.55)$$

and the variance of f on $[0, 1]$ is

$$\sigma^2 = \int_0^1 (f(x) - J(f))^2 \mu(x) dx = \int_0^1 f^2(x) dx - J^2(f). \quad (6.56)$$

The Central Limit Theorem tells us that

$$\text{prob} \left(\left| \frac{1}{n} \sum_{j=1}^n f(x_j) - J(f) \right| \leq \frac{\lambda \sigma}{\sqrt{n}} \right) \approx \frac{1}{\sqrt{2\pi}} \int_{-\lambda}^{\lambda} e^{-x^2/2} dx \quad (6.57)$$

for large n . (The Central Limit Theorem says, for large n , the sampling distribution of means with mean μ and variance σ^2 is approximately a normal distribution with mean μ and variance σ^2/n .) Thus, for example, if $\lambda = 1.96$, then

$$\text{prob} \left(\left| \frac{1}{n} \sum_{i=1}^n f(x_i) - J(f) \right| \leq \frac{1.96\sigma}{\sqrt{n}} \right) \approx 0.95.$$

That is, the error satisfies

$$E(f) = |f_n - J(f)| \leq \frac{1.96\sigma}{\sqrt{n}}$$

with probability 0.95. Hence, to reduce the error, we need to have large n or small σ . Also, notice that the error is statistical in nature and is proportional to $1/\sqrt{n}$. In the following example, the Monte Carlo approximations generally improve as n increases. In addition, notice that only a single sum is required in the approximation of the multiple integral. However, the error is still proportional to $1/\sqrt{n}$, where n is the number of points selected.

Example 6.13

$$\int_0^1 \int_0^1 \int_0^1 \int_0^1 e^{x_1 x_2 x_3 x_4} dx_1 dx_2 dx_3 dx_4 \approx 1.0693$$

n	$\frac{1}{n} \sum_{k=1}^n e^{x_{1k} x_{2k} x_{3k} x_{4k}}$
2	1.2524
16	1.0548
128	1.0574
1024	1.0742
8192	1.0696

□

There is also a variety of ways to reduce the variance σ^2 . One method is called importance sampling. Consider

$$J(f) = \int_0^1 \frac{f(x)}{p(x)} p(x) dx, \quad \text{where } p(x) > 0 \text{ and } \int_0^1 p(x) dx = 1.$$

Instead of the approximation (6.54), use

$$\bar{f}_n = \frac{1}{n} \sum_{i=1}^n \frac{f(x_i)}{p(x_i)},$$

where the x_i are random variables extracted from probability density $p(x)$ on $[0, 1]$. When we do so, the variance of f/p with respect to the density p is

$$\sigma^2 = \int_0^1 \frac{f^2(x)}{p^2(x)} p(x) dx - \left(\int_0^1 \frac{f(x)}{p(x)} p(x) dx \right)^2,$$

and if $p(x) \approx \frac{f(x)}{\int_0^1 f(x)dx}$, then $\sigma^2 \approx 0$.

Example 6.14

Let $f(x) = e^x$. Replace

$$p(x) \approx \frac{e^x}{\int_0^1 e^x dx} \quad \text{by} \quad p(x) = \frac{1+x}{\int_0^1 (1+x)dx} = \frac{2}{3}(1+x).$$

The original σ^2 is

$$\sigma^2 = \int_0^1 e^{2x} dx - \left(\int_0^1 e^x dx \right)^2 = 2e^1 - 3/2 - \frac{1}{2}e^2 \approx 0.24204.$$

The new σ^2 is

$$\sigma^2 = \frac{3}{2} \int_0^1 \frac{e^{2x}}{1+x} dx - \left(\int_0^1 e^x dx \right)^2 \approx 0.0269.$$

The new σ^2 is about a factor of 10 smaller than the original σ^2 . □

For more information on Monte Carlo methods in numerical integration see, for example, [1], [22], [28], or [55].

Quasi-Monte Carlo Methods There has been much recent interest in a new class of methods, quasi-Monte Carlo methods, which are actually deterministic rather than statistical [28, 66]. Like Monte Carlo methods, quasi-Monte Carlo methods are easily applied to multiple integrals, but generally have the advantage of higher accuracy than Monte Carlo methods. Consider a one-dimensional problem and let

$$|E_N(f)| = \left| \frac{1}{N} \sum_{n=1}^N f(x_n) - \int_0^1 f(x)dx \right|.$$

If

$$|E_N(f)| \leq c \left(\int_0^1 |f'(x)|dx \right) \frac{(\log N)^k}{N},$$

where c and k are constants, then the sequence of points $\{x_n\}_{n=1}^\infty$ is said to be a *quasi-random sequence* or a sequence of *low discrepancy* points.

Example 6.15

Consider the *Halton sequence* of low discrepancy points, as follows. Let r be a prime, and let $n = 1, 2, \dots$ be written in base r as $n = \sum_{j=0}^m a_j(n)r^j$. Then,

$\{\varphi(n)\}_{n=1}^{\infty}$ is the Halton sequence, where

$$\varphi(n) = \sum_{j=0}^n a_j(n) r^{-j-1}.$$

(Notice that the numbers are reflected about the decimal point.) For example, $r = 3$ in the following table,

n	$\sum_{j=0}^n a_j(n) r^j$	$\varphi(n)$
1	1×3^0	$1 \times 3^{-1} = 1/3$
2	2×3^0	$2 \times 3^{-1} = 2/3$
3	$0 \times 3^0 + 1 \times 3^1$	$0 \times 3^{-1} + 1 \times 3^{-2} = 1/9$
4	$1 \times 3^0 + 1 \times 3^1$	$1 \times 3^{-1} + 1 \times 3^{-2} = 4/9$
5	$2 \times 3^0 + 1 \times 3^1$	$2 \times 3^{-1} + 1 \times 3^{-2} = 7/9$

For a multidimensional (d dimensions) low discrepancy Halton sequence, let $p_1, p_2, \dots, p_d$ be the first d prime numbers. Let

$$n = \sum_{i=0}^m a_i p_j^i, \quad \text{where } a_i \in [0, p_j - 1] \text{ for } i = 0, 1, \dots, m.$$

Let

$$\varphi_{p_j}(n) = \sum_{i=0}^m a_i p_j^{-i-1}.$$

Then, $z_n = (\varphi_{p_1}(n), \varphi_{p_2}(n), \dots, \varphi_{p_d}(n))$ is the n -th d -dimensional point in the sequence. $\square$

To motivate quasi-Monte Carlo methods that use low discrepancy sequences, consider numerical integration in d dimensions. First, consider $d = 1$ and a standard numerical integration rule such as the composite trapezoidal rule. We have

$$\int_0^1 f(u) du \approx \sum_{n=0}^m w_n f\left(\frac{n}{m}\right), \quad (6.58)$$

where $w_0 = w_m = 1/(2m)$ and $w_n = 1/m$ for $1 \leq n \leq m-1$. The error in this rule is proportional to $1/m^2$, assuming that $f \in C^2[0, 1]$. Now, for general dimension d , we have

$$\int_0^1 \dots \int_0^1 f(u) du \approx \sum_{n_1=0}^m \sum_{n_2=0}^m \dots \sum_{n_d=0}^m w_{n_1} w_{n_2} \dots w_{n_d} f\left(\frac{n_1}{m}, \frac{n_2}{m}, \dots, \frac{n_d}{m}\right). \quad (6.59)$$

The total number of nodes in (6.59) is $N = (m+1)^d$. The error in (6.59) is $\mathcal{O}\left(\frac{1}{m^2}\right)$. In terms of the number of nodes, the error is $\mathcal{O}(N^{-2/d})$. Thus,

with increasing d , the approximation error increases rapidly. To guarantee a prescribed level of accuracy, say to guarantee an absolute error less than 10^{-4} , we must use roughly 10^{2d} nodes. Hence, the number of nodes increases exponentially with d , to maintain a given level of accuracy.

The Monte Carlo method helps to overcome this problem of dimensionality. The error in the Monte Carlo approximation is less than $1.96\sigma/\sqrt{N}$ with probability 0.95. Hence, the Monte Carlo method is said to have error proportional to $1/\sqrt{N}$. Notice that this error is independent of the number of dimensions and, for a large number of dimensions, Monte Carlo methods are more attractive than classical integration rules. By using low discrepancy sequences, an error bound of $\mathcal{O}((\log N)^{d-1}/N)$ is possible. Thus, quasi-Monte Carlo methods can be more accurate than Monte Carlo methods for large d .

Example 6.16

Consider $\int_0^1 5x^4 dx = 1$.

N (number of points)	Monte Carlo Estimate	Halton Estimate
500	1.18665	0.98661
1000	1.13313	0.99242
1500	1.09591	0.99375
2000	1.07040	0.99623
2500	1.03708	0.99564
3000	1.04790	0.99713
3500	1.05123	0.99741
4000	1.03659	0.99709

□

6.3.7 Adaptive Quadrature

If the function varies more rapidly in one part of the interval of integration than in other parts, and it is not known beforehand where the rapid variation is, then a single rule or a composite rule in which the subintervals all have the same length is not the most efficient. Also, in general, routines within larger numerical software libraries or packages, a user typically supplies a function f , an interval of integration $[a, b]$, and an error tolerance ϵ , without supplying any additional information about the function's smoothness.⁸ In

⁸A function is “smooth” if it has many continuous derivatives. Generally the “degree of smoothness” refers to the number of continuous derivatives available. Even if a function has, in theory, many continuous derivatives, we might consider it not to be smooth numerically if it changes curvature rapidly at certain points. An example of this is the function $f(x) = \sqrt{x^2 + \epsilon}$: as ϵ gets small, the graph of this function becomes indistinguishable from that of $f(x) = |x|$.

such cases, the quadrature routine itself should detect which portions of the interval of integration (or domain of integration in the multidimensional case) need to have a small interval length, and which portions need to have a larger interval length, to achieve the specified tolerance ϵ . In such instances, *adaptive quadrature* is appropriate.

Adaptive quadrature can be considered to be a type of *branch and bound method*.⁹ In particular, the following general procedure can be used to compute $\int_a^b f(x)dx$.

1. (Initialization)

- (a) Input an absolute error tolerance ϵ , and a minimum interval length δ .
- (b) Input the interval of integration $[a, b]$ and the function f .
- (c) $\text{sum} \leftarrow 0$.
- (d) $\mathcal{L} \leftarrow \{[a, b]\}$, where $\mathcal{L}$ is a list of subintervals that needs to be considered.

2. DO WHILE $\mathcal{L} \neq \emptyset$.

- (a) Remove the first interval from $\mathcal{L}$ and place it in the current interval $[\underline{c}, \bar{c}]$.
- (b) Apply a quadrature formula over the current interval $[\underline{c}, \bar{c}]$ to obtain an approximation I_c .
- (c) (**bound**): Use an error formula for the rule to obtain a bound E_c for the error, or else obtain E_c as a heuristic estimate for the error; This can be done by either using an error formula or by comparing with a different quadrature rule of the same or different order.
- (d) IF $E_c < \epsilon(\bar{c} - \underline{c})$, THEN

$\text{sum} \leftarrow \text{sum} + I_c$.

ELSE

IF $(\bar{c} - \underline{c}) < \delta$ THEN

RETURN with a message that the tolerance ϵ could not be met with the given minimum step size δ .

ELSE

(**branch**): form two new intervals $[\underline{c}, (\underline{c} + \bar{c})/2]$ and $[(\underline{c} + \bar{c})/2, \bar{c}]$, and store each into the list $\mathcal{L}$.

END IF

END IF

⁹We explain another type of branch and bound method, of common use in optimization, in 9.6.3 on page 523.

END DO

A good example implementation of an adaptive quadrature routine is given in the classic text [30] of Forsythe, Malcolm, and Moler.¹⁰ This routine, **quanc8**, is based on an 8-panel Newton-Cotes quadrature formula and a heuristic estimate for the error. The heuristic estimate is obtained by comparing the approximation with 8-panel rule over the entire subinterval I_c and the approximation with the composite rule obtained by applying the 8-point rule over the two halves of I_c ; see [30, pp. 94–105] for details. The routine itself¹¹ can be found in NETLIB, presently at <http://www.netlib.org/fmm/quanc8.f>.

6.3.8 Interval Bounds

Mathematically rigorous bounds on integrals can be computed, for simple integration, for composite rules, and for adaptive quadrature, if interval arithmetic is used in the error formulas. As an example, take the two point Gauss–Legendre quadrature rule:

$$\int_{-1}^1 f(x)dx = \left\{ f\left(\frac{-1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right) \right\} + \frac{1}{135}f^{(4)}(\xi), \quad (6.60)$$

for some $\xi \in [-1, 1]$, where the quadrature formula is obtained from Table 6.1 (on page 349) and where the error term is obtained from Formula (6.38) with $a = -1$, $b = 1$, and

$$\int_{-1}^1 p_{m+1}^2(x)dx = \int_{-1}^1 \left(x^2 - \frac{1}{3}\right)^2 dx = \frac{8}{45}.$$

Now, suppose we want to find guaranteed error bounds on the integral

$$\int_{-1}^1 e^{0.1x} dx.$$

Then, the fourth derivative of $e^{0.1x}$ is $(.1)^4 e^{0.1x}$, and an interval evaluation of this over $x = [-1, 1]$ gives

$$(0.1)^4 e^{0.1x} \in [0.9048, 1.1052] \times 10^{-4} \quad \text{for } x \in [-1, 1],$$

where the interval enclosure for the range $e^{0.1[-1,1]}$ was obtained using the MATLAB toolbox INTLAB [77]. The application of the quadrature rule thus

¹⁰This text doubles as an elementary numerical analysis text and as a “user guide” for the routines it explains. It distinguished itself from other texts of the time by featuring routines that were simple enough to be used to explain the elementary concepts, yet sophisticated enough to be used to solve practical problems.

¹¹In Fortran 66, but written carefully and clearly.

gives

$$\begin{aligned}\int_{-1}^1 e^{0.1x} dx &\in e^{-0.1/\sqrt{3}} + e^{0.1/\sqrt{3}} + [0.9048, 1.1052] \times 10^{-4} \\ &\subseteq e^{-0.1/[1.7320, 1.7321]} + e^{0.1/[1.7320, 1.7321]} + [0.9048, 1.1052] \times 10^{-4} \\ &\subseteq [2.0034, 2.0035],\end{aligned}$$

where the computations were done within INTLAB. This set of computations provides a mathematical proof that the exact integral lies within $[2.0034, 2.0035]$.

The higher order derivatives required in the quadrature formulas can be bounded over intervals using a combination of automatic differentiation (explained in §6.2, starting on page 327, of this book) and interval arithmetic.

The mathematically rigorous error bounds obtained by this technique can be used in an adaptive quadrature technique, and the resulting routine can give mathematically rigorous bounds, provided I_c and `sum` are computed with interval arithmetic and the error bounds are added to each I_c when it is added to `sum`. Such a routine is described in [21], although an updated package was not widely available at the time this book was written.

6.4 Exercises

1. Carry out the details of the computation to derive (6.60).
2. Assume that we have a finite-difference approximation method where the roundoff error is $\mathcal{O}(\epsilon/h)$ and the truncation error is $\mathcal{O}(h^n)$. Using the error bounding technique exemplified in (6.10) on page 326, show that the optimal h is $\mathcal{O}(\epsilon^{1/(n+1)})$ and the minimum achievable error bound is $\mathcal{O}(\epsilon^{n/(n+1)})$.
3. Consider the finite difference formula (6.8).
 - (a) Derive an error bound (bounding both roundoff and truncation) for this formula analogous to (6.10).
 - (b) Compute an optimal h and a minimal achievable error, as was done following (6.10).
 - (c) Produce a table similar to that in Example 6.2, but using this formula.
 - (d) Compare the results in your table to the optimal h and minimal achievable error you have computed.
4. Repeat Exercise 3, but for formula (6.9) instead of formula (6.8).

5. Assume that $f \in C^3[a, b]$ and $x_0, x_0 + h, x_0 + 2h \in [a, b]$. Prove that there exist constants c_1 and c_2 such that

$$\left| f'(x_0) - \frac{1}{h} \left[-\frac{3}{2}f(x_0) + c_1f(x_0 + h) + c_2f(x_0 + 2h) \right] \right| \leq c h^2 \max_{a \leq x \leq b} |f'''(x)|,$$

where $c > 0$ is a constant independent of h .

6. Let $f \in C^\infty(-\infty, \infty)$ and let $x_0 \in \mathbb{R}$ be given.

- (a) Prove that $C_h = \frac{f(x_0 + h) - f(x_0 - h)}{2h} = f'(x_0) + \sum_{i=1}^{\infty} c_i h^{2i}$, where $c_i, i = 1, 2, 3, \dots$ are independent of h .
- (b) Suppose that C_h and $C_{\frac{h}{2}}$ have been calculated. Find constants α_1 and α_2 so that

$$\alpha_1 C_h + \alpha_2 C_{\frac{h}{2}} = f'(x_0) + O(h^4).$$

7. Write down a general formula for the $(k + 1)$ -st component of $\sin(u_{\nabla})$, as in Formula (6.14) on page 329. *Note: It is permissible to look this up, in this case.*
8. Write down a general formula for the $(k + 1)$ -st component of $(u_{\nabla})^n$, following the forward differentiation scheme of §6.2.1. *Note: as with Exercise 7, it is permissible to look this up, in this case.*
9. Complete the computations for Example 6.4.
10. Solve the system (6.19) (on page 333) and compare your result to the corresponding directional derivative of f computed by taking the gradient of f and taking the dot product with the direction.
11. Suppose $Q(f)$ is any integration rule of the form (6.21) (page 334) that integrates constants exactly, that is, such that

$$\int_a^b 1 dx - Q(1) = 0.$$

Further, assume that $f \in C[a, b]$, and apply the composite quadrature rule (6.23) (on page 335) based on Q . Prove that

$$E(f) = \int_a^b f(x) dx - \sum_{\nu=0}^{N-1} Q_{\nu}(f) \rightarrow 0 \quad \text{as } N \rightarrow \infty.$$

12. Let $Q_m(f)$ denote the m -point Gaussian quadrature rule over the interval $[a, b]$ and with continuous weight function ρ , that is,

$$Q_m(f) = \sum_{i=1}^m \alpha_i f(x_i) \approx J(f) = \int_a^b \rho(x) f(x) dx.$$

Show that, if a and b are finite and f is continuous, then $Q_m(f) \rightarrow J(f)$ as $m \rightarrow \infty$.

(Hint: You may wish to consider the Weierstrass approximation theorem, page 205, as well as Theorem 6.3 on page 352.)

13. Assume that $f \in C^2[a, b]$. Let $M = \max_{x \in [a, b]} |f''(x)|$.

(a) Prove that $\left| \int_a^b f(x) dx - (b-a)f\left(\frac{a+b}{2}\right) \right| \leq (b-a)^3 \frac{M}{24}$

(b) $\left| \int_a^b f(x) dx - \sum_{j=0}^{N-1} \frac{b-a}{N} f\left(\frac{a_j + a_{j+1}}{2}\right) \right| \leq (b-a)^3 \frac{M}{24N^2}$

14. Let $f \in C[0, 1]$ and let $\sum_{i=1}^n w_i f(x_i)$ be an approximation of $\int_0^1 f(x) dx$.

Assume that $0 \leq w_i \leq 1$ and $0 \leq x_i \leq 1$, $i = 1, 2, \dots, n$ and $\sum_{i=1}^n w_i = 1$ for any n . Let $E_n(f) = \int_0^1 f(x) dx - \sum_{i=1}^n w_i f(x_i)$ be the error in the approximation. Assume that $E_n(P_n) = 0$ for any P_n , a polynomial of degree $\leq n$. Prove that given $\epsilon > 0$ there exists an $N > 0$ such that $|E_n(f)| \leq \epsilon$ for all $n \geq N$.

15. Consider the integral approximation: $\int_{-1}^1 f(x) dx \approx f(-a) + f(a)$

(a) Prove that the error in this approximation is bounded by

$$(a^3 - a^2 + \frac{1}{3}) \max_{x \in [-1, 1]} |f''(x)|.$$

(b) What is the optimal a in the sense that the integration rule is exact for the highest degree polynomial possible?

16. Answer the following:

(a) Consider the formula

$$\int_0^h f(x) dx = h \left\{ Af(0) + Bf\left(\frac{h}{3}\right) + Cf(h) \right\}.$$

Find A , B , and C such that this is exact for all polynomials of degree less than or equal to 2.

- (b) Suppose that the Trapezoidal rule applied to $\int_0^2 f(x)dx$ gives the value $\frac{1}{2}$ while the quadrature rule in part (a) applied to $\int_0^2 f(x)dx$ gives the value $\frac{1}{4}$. If $f(0) = 3$, then show that $f(\frac{2}{3}) = 1$.

17. Let $\int_0^1 f(x) dx \approx \sum_{i=0}^{N-1} f(x_i)h$ where $x_i = ih$ and $h = \frac{1}{N}$. Suppose that $f \in C^1[0, 1]$. Prove that $\left| \int_0^1 f(x) dx - \sum_{i=0}^{N-1} f(x_i) h \right| \leq \frac{h}{2} \max_{0 \leq x \leq 1} |f'(x)|$.

18. Consider the quadrature formula of the type

$$\int_0^1 f(x) [x \ln(1/x)] dx = a_0 f(0) + a_1 f(1).$$

- (a) Find a_0 and a_1 such that the formula is exact for linear polynomials.

- (b) Describe how the above formula, for $h > 0$, can be used to approximate $\int_0^h g(t) t \ln(h/t) dt$.

19. Suppose that $I(h)$ is an approximation to $\int_a^b f(x) dx$, where h is the width of a uniform subdivision of $[a, b]$. Suppose that the error satisfies

$$I(h) - \int_a^b f(x) dx = c_1 h + c_2 h^2 + \mathcal{O}(h^3),$$

where c_1 and c_2 are constants independent of h . Let $I(h)$, $I(h/2)$, and $I(h/3)$ be calculated for a given value of h . Use the values $I(h)$, $I(h/2)$ and $I(h/3)$ to find an $\mathcal{O}(h^3)$ approximation to $\int_a^b f(x) dx$.

20. Use a two point Gauss quadrature rule to approximate the following integral:

$$I = \int_{-\infty}^1 \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx.$$

21. Find the nodes x_i and the corresponding weights A_i , $i = 0, 1, 2$, so the formula

$$\int_{-1}^1 \frac{1}{\sqrt{1-x^2}} f(x) dx \approx \sum_{i=0}^2 A_i f(x_i)$$

is exact when $f(x)$ is any polynomial of degree 5. Compare your solution with the roots of the Chebyshev polynomial of the first kind T_3 , given by $T_3(x) = \cos(3 \cos^{-1}(x))$.

22. Let $\Phi(x)$ be the piecewise linear interpolant of $f(x)$ on the partition $a = x_0 < x_1 < \dots < x_n = b$, where $x_j = a + jh$ for $j = 0, 1, \dots, n$ and $h = \frac{b-a}{n}$. Show that $\int_a^b \Phi(x) dx$ is equivalent to the composite trapezoidal rule approximation to $\int_a^b f(x) dx$.
23. Let $f \in C^1([0, 1] \times [0, 1])$ and $h = \frac{1}{n}$. Show that

$$\left| \int_0^1 \int_0^1 f(x, y) dx dy - h^2 \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} f(ih, jh) \right| \leq ch$$

for some constant c independent of h . (Recall

$$f(x, y) = f(a, b) + \frac{\partial f}{\partial x}(\mu, \xi)(x - a) + \frac{\partial f}{\partial y}(\mu, \xi)(y - b)$$

for some (μ, ξ) on the line between (x, y) and (a, b) .)

24. Suppose that a particular composite quadrature rule is used to approximate $\int_0^2 e^{x^2} dx$. The following values are obtained for $N = 8, 16$, and 32 intervals, respectively: 16.50606, 16.45436, and 16.45347. Using only these values, estimate the power of h to which the error is proportional, where $h = \frac{2}{N}$.
25. A two dimensional Gaussian quadrature formula has the form

$$\int_{-1}^1 \int_{-1}^1 f(x, y) dx dy = f(\alpha, \alpha) + f(-\alpha, \alpha) + f(\alpha, -\alpha) + f(-\alpha, -\alpha) + E(f).$$

Find the value of α such that the formula is exact (i.e. $E(f) = 0$) for every polynomial $f(x, y)$ of degree less than or equal to 2 in 2 variables

$$\text{i.e., } f(x, y) = \sum_{i,j=0}^2 a_{ij} x^i y^j.$$

26. Consider approximation $J(f) = \int_0^1 e^x dx$ using the Monte Carlo method. Find the number of points n so that

$$\text{prob} \left(\left| \frac{1}{n} \sum_{i=1}^n e^{x_i} - \int_0^1 e^x dx \right| \leq 0.001 \right) = 0.997.$$

Chapter 7

Initial Value Problems for Ordinary Differential Equations

7.1 Introduction

In this chapter, we are interested in approximating the solution of the following *initial-value problem* (IVP) for a first-order system of differential equations. We seek to approximate $y : [a, b] \rightarrow \mathbb{R}^n$ that satisfies

$$\begin{cases} y'(t) = f(t, y(t)), & a \leq t \leq b, \\ y(a) = y_0, \end{cases} \quad (7.1)$$

where f is a given $\mathbb{R}^n$ -valued function of $n + 1$ real variables and y_0 is a given vector in $\mathbb{R}^n$. That is, we seek n functions $y_i(t)$, $1 \leq i \leq n$, defined for $a \leq t \leq b$ such that

$$\begin{cases} y'_i(t) = f_i(t, y_1(t), y_2(t), \dots, y_n(t)), & 1 \leq i \leq n, & a \leq t \leq b, \\ y_i(a) = y_{0,i}. \end{cases}$$

For example, for $n = 2$,

$$\begin{cases} y'_1(t) = \cos t + y_1(t)y_2(t) = f_1(t, y_1, y_2) \\ y'_2(t) = y_2(t) - y_1(t) = f_2(t, y_1, y_2) \\ y_1(1) = 2, \quad y_2(1) = 0. \end{cases}$$

However, for a general function f , problem (7.1) need not have a solution on the interval $[a, b]$. For example, in $\mathbb{R}^1$, consider

$$y'(t) = |y(t)|^{3/2}, \quad 0 \leq t \leq 3, \quad y(0) = 1.$$

The function $y(t) = (1 - t/2)^{-2}$ is a solution to this IVP on the interval $[0, b]$, $0 < b < 2$; this solution blows up as $t \rightarrow 2$. In fact, a solution to this IVP does not exist over the entire interval $[0, 3]$.

Additionally, even if a solution to the IVP exists over all of $[a, b]$, the solution may not be unique. As an example of this phenomenon, consider the IVP

$$y'(t) = |y(t)|^{1/2}, \quad 0 \leq t \leq 1, \quad y(0) = 0.$$

Clearly, $y(t) = 0$, $0 \leq t \leq 1$ is a solution. Also, it is easily shown that

$$y(t) = \begin{cases} 0, & 0 \leq t \leq 1/2, \\ (t - 1/2)^2/4, & 1/2 \leq t \leq 1 \end{cases}$$

is also a solution. Hence, solutions to this IVP are not unique on $[0, 1]$.

Problem (7.1) has a unique solution in some interval¹ about $t = a$ if $f_1, f_2, \dots, f_n$ are continuous and possess continuous first partial derivatives as stated in the following theorem. Details and a proof can be found in many books on differential equations, such as [94].

THEOREM 7.1

(local existence and uniqueness) If there exists a positive number γ such that $f_1, f_2, \dots, f_n$ are continuous and possess continuous first partial derivatives with respect to the components $y_1, y_2, \dots, y_n$ of y , for

$$|t - a| < \gamma, |y_1 - y_{0,1}| < \gamma, |y_2 - y_{0,2}| < \gamma, \dots, |y_n - y_{0,n}| < \gamma, \quad (7.2)$$

then there exists an $\eta > 0$ such that the system (7.1) has a unique solution for $|t - a| \leq \eta$.

Nonlocal existence is guaranteed by the following theorem. A classic reference for Theorem 7.2 is [18].

THEOREM 7.2

Let f be a continuous vector-valued function² defined on

$$S = \{(t, y) \mid t \in \mathbb{R}, y \in \mathbb{R}^n, |t - a| \leq \gamma, \|y\| < \infty\},$$

and let f satisfy a Lipschitz condition in the y variable over S . Then $y' = f(t, y)$, $y(a) = y_0$ has a unique solution for $|t - a| \leq \gamma$.

Analogously to Definition 2.2 (page 40), we have

DEFINITION 7.1 *f satisfies a Lipschitz condition with respect to y provided*

$$\|f(t, y) - f(t, \tilde{y})\| \leq L\|y - \tilde{y}\|, \quad (7.3)$$

where L is a constant (Lipschitz constant) independent of $y, \tilde{y}$, and t , and $\|\cdot\|$ is some norm in $\mathbb{R}^n$.

¹not necessarily a prespecified interval $[a, b]$

²That is, $f(t, y) = (f_1(t, y), \dots, f_n(t, y))^T$.

REMARK 7.1 Note that if f is Lipschitz continuous with respect to one norm in $\mathbb{R}^n$, then, by equivalence of vector norms, it is Lipschitz continuous with respect to any other norm in $\mathbb{R}^n$ but with a different Lipschitz constant. In addition, if each $\partial f_i / \partial y_j$, $1 \leq i \leq n$, $1 \leq j \leq n$ is continuous and bounded on S , then application of the mean value theorem in componentwise form yields

$$f_i(t, y) - f_i(t, \tilde{y}) = \sum_{j=1}^n \frac{\partial f_i(t, c_i)}{\partial y_j} (y_j - \tilde{y}_j)$$

for some point $c_i \in \mathbb{R}^n$ on the line segment in $\mathbb{R}^n$ between y and $\tilde{y}$. (This follows from the multivariate mean value theorem, stated as Theorem 8.1 on page 441 and proven in Exercise 1 on page 482 in Section 8.1). By letting

$$L = \max_{t, y \in S} \max_i \sum_{j=1}^n \left| \frac{\partial f_i(t, y)}{\partial y_j} \right|,$$

the multivariate mean value theorem gives condition (7.3) for $\|\cdot\| = \|\cdot\|_\infty$. $\square$

REMARK 7.2 Traditionally,³ IVP's for higher order differential equations are not considered separately from first-order equations. By a change of variables, higher order problems can be reduced to a system of the form of (7.1). For example, consider the scalar IVP for the m -th-order scalar differential equation:

$$\begin{cases} y^{(m)}(t) = g(t, y^{(m-1)}(t), y^{(m-2)}(t), \dots, y''(t), y'(t), y(t)), & a \leq t \leq b, \\ y(a) = u_0, y'(a) = u_1, \dots, y^{(m-1)}(a) = u_{m-1}. \end{cases} \quad (7.4)$$

We can reduce this high order IVP to a first-order system of the form (7.1) by defining $x : [a, b] \rightarrow \mathbb{R}^m$ componentwise by

$$x(t) = [x_1(t), x_2(t), \dots, x_m(t)]^T = [y(t), y'(t), y^{(2)}(t), \dots, y^{(m-1)}(t)]^T.$$

Then,

$$\begin{cases} x'_1(t) = x_2(t), \\ x'_2(t) = x_3(t), \\ x'_3(t) = x_4(t), \\ \vdots \\ x'_{m-1}(t) = x_m(t), \\ x'_m(t) = g(t, x_m, x_{m-1}, \dots, x_2, x_1), \end{cases} \quad \text{with } x(a) = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{m-1} \end{bmatrix}. \quad (7.5)$$

³Recently, there has been some discussion concerning efficient methods that do consider higher-order problems separately.

That is, in this case $f(t, x)$ is defined by:

$$\begin{cases} f_i(t, x) = x_{i+1}, & 1 \leq i \leq m-1 \\ f_m(t, x) = g(t, x_m, x_{m-1}, \dots, x_2, x_1). \end{cases}$$

□

Example 7.1

Consider

$$\begin{cases} y''(t) = y'(t) \cos(y(t)) + e^{-t}, \\ y(0) = 1, \\ y'(0) = 2. \end{cases}$$

Let $x_1 = y$ and $x_2 = y'$. Then,

$$\begin{cases} x'_1(t) = x_2(t), \\ x'_2(t) = x_2(t) \cos(x_1(t)) + e^{-t}, \\ x_1(0) = 1, x_2(0) = 2, \end{cases}$$

which can be represented in vector form as

$$\begin{cases} \frac{dx}{dt} = f(t, x) = \begin{bmatrix} x_2(t) \\ x_2(t) \cos(x_1(t)) + e^{-t} \end{bmatrix}, \\ x(0) = \begin{bmatrix} 1 \\ 2 \end{bmatrix}. \end{cases}$$

□

In this chapter, we are concerned with numerical solution of (7.1), i.e., we shall seek approximations to $y(t)$, the solution of (7.1), at a discrete set of points $t_0, t_1, \dots, t_N \in [a, b]$, where N is a positive integer. Although in practice the t_k 's need not be equally spaced,⁴ we will assume here that they are to simplify the presentation of the theoretical analysis. With that assumption, we define the *step size* or *step* or *mesh length* to be

$$h = \frac{b-a}{N}.$$

The nodes of the numerical scheme will be defined to be the equidistant points

$t_k = a + kh$, $0 \leq k \leq N$, i.e. $t_0 = a$, $t_1 = a + h$, $\dots$, $t_N = b$ and $t_{k+1} - t_k = h$.

A numerical scheme will produce vectors $y_0, y_1, \dots, y_N$, which will approximate the solution $y(t)$ at $t = t_0 = a$, $t = t_1$, $t = t_2$, $\dots$, $t = t_N = b$,

⁴Modern software for differential equations, besides using methods in this chapter, employs *adaptive techniques* to adjust the distance between points, to achieve a required accuracy without excessive work.

respectively, i.e., $y_k \approx y(t_k)$. (Henceforth, in this chapter, y_k will denote an element of a sequence of vectors in $\mathbb{R}^n$, as opposed to the k -th component of a vector y . Similarly, f_k will be a vector in $\mathbb{R}^n$ that denotes the k -th element of a sequence of values of f , as opposed to the value of the k -th component of f .)

7.2 Euler's method

The simplest method we consider is *Euler's method* (also called the Cauchy polygon method), given by the following iteration scheme.

$$\begin{cases} y_0 = y(a), \\ y_{k+1} = y_k + hf(t_k, y_k), \quad 0 \leq k \leq N-1. \end{cases} \quad (7.6)$$

There are many ways of deriving (7.6). For example, approximating $y'(t)$ by the forward difference formula, we obtain

$$\frac{y(t_{k+1}) - y(t_k)}{h} \approx y'(t_k) = f(t_k, y(t_k)).$$

Hence, $y(t_{k+1}) \approx y(t_k) + hf(t_k, y(t_k))$, which immediately suggests (7.6).

Example 7.2

Consider the scalar, i.e. $n = 1$, problem:

$$\begin{cases} y'(t) = t + y, \\ y(1) = 2. \end{cases}$$

(The exact solution is $y(t) = -t - 1 + 4e^{-1}e^t$.) Euler's method has the form

$$\begin{cases} y_{k+1} = y_k + hf(t_k, y_k) = y_k + h(t_k + y_k), \\ y_0 = 2, \quad t_0 = 1. \end{cases}$$

Applying Euler's method, we obtain

$h = 0.1$				$h = 0.05$			
k	t_k	y_k	$y(t_k)$				
0	1	2	2				
1	1.1	2.3	2.32068				
2	1.2	2.64	2.68561				
3	1.3	3.024	3.09944				
4	1.4	3.4564	3.56730				
5	1.5	3.94304	4.09489				

k	t_k	y_k	$y(t_k)$
0	1	2	2
1	1.05	2.1500	2.15508
2	1.1	2.3100	2.32068

The error for $h = 0.1$ at $t = 1.1$ is about 0.02 and the error for $h = 0.05$ at $t = 1.1$ is about 0.01. If h is cut in half, the error is cut in half, suggesting that the error is proportional to h . $\square$

We will now study the convergence of the sequence $\{y_k\}_{0 \leq k \leq N} \subset \mathbb{R}^n$ to the solution $y(t)$, $a \leq t \leq b$, where $y_k \approx y(t_k) = y(a + kh) = y(a + k(b - a)/N)$. We require the following lemma:

LEMMA 7.1

Suppose that there exist positive constants δ and K^* such that the members of the sequence $d_0, d_1, \dots$ satisfy

$$d_{n+1} \leq d_n (1 + \delta) + K^*, \quad n = 0, 1, 2, \dots \quad (7.7)$$

Then,

$$d_n \leq d_0 e^{n\delta} + \frac{K^*}{\delta} (e^{n\delta} - 1), \quad n = 0, 1, 2, \dots \quad (7.8)$$

PROOF By recursively applying (7.7),

$$\begin{aligned} d_n &\leq (1 + \delta)^n d_0 + K^* (1 + (1 + \delta) + (1 + \delta)^2 + \dots + (1 + \delta)^{n-1}) \\ &= (1 + \delta)^n d_0 + K^* \frac{((1 + \delta)^n - 1)}{\delta} \\ &\leq e^{n\delta} d_0 + K^* \frac{e^{n\delta} - 1}{\delta} \end{aligned}$$

(after noting that $1 + \delta \leq e^\delta = 1 + \delta + \frac{\delta^2}{2} + \dots$). $\square$

An *error estimate*, i.e., a bound on $y_j - y(t_j)$, will now be obtained in the norm $\|\cdot\|$ in which the Lipschitz condition (7.3) holds. For this norm, we suppose that

$$c_2 \|x\|_\infty \leq \|x\| \leq c_1 \|x\|_\infty; \quad \|x\|_\infty = \max_{1 \leq i \leq n} |x_i| \quad (7.9)$$

for some c_1 and c_2 independent of x , since any two norms on $\mathbb{R}^n$ are equivalent. We have the following result:

THEOREM 7.3

Suppose that $f(t, y)$ satisfies the conditions of Theorem 7.2, so that there is a unique solution $y(t)$ of (7.1) for $a \leq t \leq b$. Let L be the Lipschitz constant defined in (7.3), and let $c_1 > 0$ be the constant defined in (7.9). Suppose that $y''(t)$ exists and is continuous for $a \leq t \leq b$ and that

$$\max_{a \leq t \leq b} \|y''(t)\|_\infty = M. \quad (7.10)$$

Let $\{y_k\}_{k=1}^N$ be the discrete solution generated by the Euler method (7.6). Then we have the following error estimate.

$$\max_{1 \leq k \leq N} \|y_k - y(t_k)\| \leq h \frac{c_1 M}{2L} [e^{L(b-a)} - 1]. \quad (7.11)$$

(Note that the error is $\mathcal{O}(h)$.)

PROOF We define the *discretization error* or *global truncation error* at step k by

$$e_k = y_k - y(t_k), \quad 0 \leq k \leq N. \quad (7.12)$$

If we assume the initial condition $y_0 = y(t_0)$ is exact, then $e_0 = 0$. Then, Euler's method gives

$$y_{k+1} = y_k + hf(t_k, y_k), \quad 0 \leq k \leq N-1. \quad (7.13)$$

Consider now the solution $y(t) = [y_1(t), y_2(t), \dots, y_n(t)]^T$. (Here, the subscript " i " denotes a vector component, rather than an iterate.) By Taylor's theorem,

$$y_i(t_{k+1}) = y_i(t_k) + hy'_i(t_k) + \frac{h^2}{2} y''_i(\xi_i^k) \quad (7.14)$$

for each i , $1 \leq i \leq n$, where ξ_i^k is some point in the interval $[t_k, t_{k+1}]$. Denoting by v_k the vector with components $v_i^k = y''_i(\xi_i^k)$, we write (7.14) as

$$y(t_{k+1}) = y(t_k) + hf(t_k, y(t_k)) + \frac{h^2}{2} v_k. \quad (7.15)$$

Note by (7.9) and (7.10) that

$$\|v_k\| \leq c_1 \|v_k\|_\infty \leq c_1 M \quad (7.16)$$

for $k = 0, 1, \dots, N$. Subtracting (7.15) from (7.13) and using definition (7.12) we have

$$e_{k+1} = e_k + h[f(t_k, y_k) - f(t_k, y(t_k))] - \frac{h^2}{2} v_k. \quad (7.17)$$

Taking the norm of each side and using (7.16) gives

$$\|e_{k+1}\| \leq \|e_k\| + h\|f(t_k, y_k) - f(t_k, y(t_k))\| + \frac{h^2}{2} c_1 M. \quad (7.18)$$

Using the Lipschitz condition (7.3), we obtain

$$\|e_{k+1}\| \leq (1 + hL)\|e_k\| + \frac{h^2}{2} c_1 M, \quad 0 \leq k \leq N-1, \quad (7.19)$$

with $\|e_0\| = 0$. Hence, using Lemma 7.1, by setting $d_k = \|e_k\|$, $\delta = hL$, $K^* = h^2 c_1 M/2$, and using $kh \leq b-a$ for $0 \leq k \leq N$, we obtain

$$\|e_k\| \leq h \frac{c_1 M}{2L} [e^{L(b-a)} - 1]. \quad (7.20)$$

□

REMARK 7.3 Note that the right side of (7.20) is of the form ch , where c is a constant independent of h . Thus, as $h \rightarrow 0$, $y_k \rightarrow y(t_k)$, i.e., Euler's method is *convergent*. It can be shown that the error estimate above is the best in the sense that the error is not of the form, for example, ch^2 . In practice, the actual errors $\|e_k\|$ of Euler's method are usually smaller than that predicted by (7.20), i.e., the constant $\frac{c_1 M}{2L} [e^{L(b-a)} - 1]$ is pessimistic. However, the linear behavior of the error is apparent, i.e., if the step size h is cut in half, then the error is generally reduced by about a factor of $\frac{1}{2}$. □

REMARK 7.4 We have assumed so far that the calculations in Euler's method have been performed exactly. Consider now the effects of rounding errors. First, assume that

$$\tilde{y}_0 = y_0 + e_0, \quad (7.21)$$

where we think of $e_0 \in \mathbb{R}^n$ as being “very small.” (Note that e_0 represents error in representation of the initial condition in the machine.) Second, assume that the rounding error in the calculation of f is

$$\tilde{f}(t_k, \tilde{y}_k) = f(t_k, \tilde{y}_k) + \epsilon_k, \quad \text{where } \epsilon_k \text{ represents roundoff error.} \quad (7.22)$$

Finally, assume rounding errors occur when we multiply $\tilde{f}$ by h and add to $\tilde{y}_k$, i.e.,

$$\tilde{y}_{k+1} = \tilde{y}_k + h\tilde{f}(t_k, \tilde{y}_k) + \rho_k, \quad \text{where } \rho_k \text{ represents roundoff error.} \quad (7.23)$$

If we assume that, for all k ,

$$\|\epsilon_k\| \leq \epsilon \quad \text{and} \quad \|\rho_k\| \leq \rho,$$

then it can be shown⁵ that

$$\begin{aligned} \max_{0 \leq k \leq N} \|\tilde{y}_k - y(t_k)\| &\leq c_1 \|e_0\| e^{L(b-a)} \\ &\quad + c_1 \left(\frac{e^{L(b-a)} - 1}{L} \right) \left(\frac{hM}{2} + \epsilon + \frac{\rho}{h} \right). \end{aligned} \quad (7.24)$$

REMARK 7.5 If $\|e_0\| = \epsilon = \rho = 0$, then (7.24) reduces to (7.11). Also, note that $e_0 = 0$ if y_0 is exactly representable in the machine. □

What is interesting about the bound (7.24) is that the rounding error becomes unbounded as $h \rightarrow 0$, i.e., as the number of intervals, and hence the number of calculations, goes to infinity. Indeed if we plot the total error, we obtain the graph of Figure 7.1. Thus, if we wish very accurate calculations,

⁵in a manner very similar to proof of Theorem 7.3; you will do this in Exercise 1 on page 431.

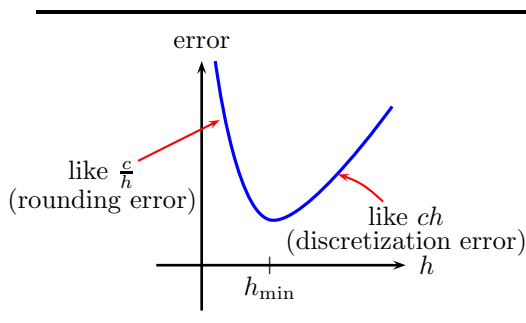


FIGURE 7.1: Illustration of rounding plus discretization error (Remark 7.5).

then we may have to use extended precision⁶ to reduce the constants ϵ , ρ , and δ . Of course, this increases the cost of the method. In subsequent sections, we shall seek more accurate methods, i.e., with error bounds of the form ch^2 , ch^3 , ch^4 , etc. However, the general picture of *discretization error* versus *rounding error* persists in these methods. $\square$

7.3 Single-Step Methods: Taylor Series and Runge–Kutta

The objective of this section is to derive higher order methods, i.e., schemes whose errors are $\mathcal{O}(h^k)$, $k > 1$. In particular, we shall consider explicit Taylor series and Runge–Kutta methods, which form a basic class of *explicit single-step methods* for numerical solution of (7.1).

An explicit single-step method (or one-step) method for numerical solution of (7.1) is a method of the form

$$\begin{cases} y_0 = y(t_0) \\ y_{k+1} = y_k + h\Phi(t_k, y_k, h), \quad 0 \leq k \leq N-1, \end{cases} \quad (7.25)$$

⁶At the time of the writing of this work, “standard” precisions correspond to IEEE 754 single precision and IEEE 754 double precision, as given in Table 1.1 on page 19. Extended precision corresponds to more bits in the mantissa than one of these two “standard” precisions. For example, IEEE double precision uses 64 binary digits (bits) to represent the number, while IEEE quadruple precision, considered an “extended precision” uses 128 bits total. Computer chips are typically designed so that single and double precision computations are much faster than extended precision.

where Φ is a given $\mathbb{R}^n$ -valued function defined on $[a, b] \times \mathbb{R}^n \times [0, h_0]$, for some constant $h_0 > 0$.

The first example of a one-step method is Euler's method (7.6), for which $\Phi(t_k, y_k, h) = f(t_k, y_k)$. Let's derive another one-step method.

For simplicity, consider $n = 1$. If $y(t)$ is the exact solution of (7.1), then

$$y(t_{k+1}) - y(t_k) = \int_{t_k}^{t_{k+1}} f(t, y(t)) dt, \quad 0 \leq k \leq N-1. \quad (7.26)$$

Approximating the integral on the right side by the midpoint rule, we obtain

$$\int_{t_k}^{t_{k+1}} f(t, y(t)) dt \approx hf \left(t_k + \frac{h}{2}, y(t_k + \frac{h}{2}) \right). \quad (7.27)$$

Now, by Taylor's Theorem,

$$y(t_k + \frac{h}{2}) \approx y(t_k) + \frac{h}{2} y'(t_k) = y(t_k) + \frac{h}{2} f(t_k, y(t_k)). \quad (7.28)$$

By (7.26), (7.27), and (7.28), it is seen that $y(t)$ approximately satisfies

$$\begin{cases} y(t_{k+1}) \approx y(t_k) + hf \left(t_k + \frac{h}{2}, K_1 \right), & 0 \leq k \leq N-1, \\ \text{with } K_1 = y(t_k) + \frac{h}{2} f(t_k, y(t_k)), \end{cases} \quad (7.29)$$

which suggests the following numerical method, known as the *midpoint method* for solution of (7.1). We seek y_k , $0 \leq k \leq N$, such that

$$\begin{cases} y_0 = y(t_0), \\ y_{j+1} = y_j + hf \left(t_j + \frac{h}{2}, K_{1,j} \right), & j = 0, 1, 2, \dots, N-1, \\ K_{1,j} = y_j + \frac{h}{2} f(t_j, y_j). \end{cases} \quad (7.30)$$

We can write (7.30) in the form:

$$\begin{cases} y_0 = y(t_0) \\ y_{j+1} = y_j + h\Phi(t_j, y_j, h), \end{cases} \quad (7.31)$$

where

$$\Phi(t_j, y_j, h) = f(t_j + \frac{h}{2}, y_j + \frac{h}{2} f(t_j, y_j)).$$

Hence, the midpoint method is of the form (7.25). In addition, it is a Runge-Kutta method as defined later. Before continuing, some definitions are introduced.

DEFINITION 7.2 We say that a single-step method (7.25) has order of accuracy p , provided p is the largest integer for which

$$y(t+h) - [y(t) + h\Phi(t, y, h)] = \mathcal{O}(h^{p+1}), \quad (7.32)$$

where $y(t)$ is the exact solution of IVP (7.1).

DEFINITION 7.3 A single-step method (7.25) is consistent with the differential equation $y'(t) = f(t, y)$ if $\Phi(t, y, 0) = f(t, y)$.

We will see later that these definitions are useful. Let us investigate the order and consistency of Euler's method and the midpoint method.

7.3.1 Order and Consistency of Euler's Method

Euler's method is consistent, since, clearly, $\Phi(t, y, 0) = f(t, y)$. Now let $y(t)$ be the solution of (7.1). Then

$$y(t+h) - [y(t) + h\Phi(t, y, h)] = y(t+h) - y(t) - hf(t, y(t)). \quad (7.33)$$

Applying Taylor's Theorem, for $1 \leq i \leq n$,

$$y_i(t+h) = y_i(t) + hy'_i(t) + \frac{h^2}{2}y''_i(\xi_i), \quad \xi_i \in [t, t+h], \quad (7.34)$$

where, here, y_i is the i -th component of y . Suppose that $y''(t)$ is continuous for $a \leq t \leq b$ and let

$$\max_{a \leq t \leq b} \|y''(t)\|_\infty = M. \quad (7.35)$$

By (7.33), (7.34), and using $y'(t) = f(t, y(t))$, we obtain

$$\|y(t+h) - [y(t) + h\Phi(t, y, h)]\|_\infty \leq \frac{h^2}{2}M. \quad (7.36)$$

Thus, by Definition 7.2, Euler's method has order of accuracy 1.

7.3.2 Order and Consistency of the Midpoint Method

For simplicity, let's consider the scalar case $n = 1$. The midpoint method is consistent since, clearly, $\Phi(t, y, 0) = f(t, y)$.

For notational convenience, define

$$d(y(t), h) = y(t+h) - y(t) - h\Phi(t, y(t), h), \quad (7.37)$$

where $y(t)$ solves (7.1) for $n = 1$. The quantity $d(y(t), h)$ is often called the *local truncation error*, and is a measure of the amount that the exact solution fails to satisfy the difference equation (approximate method) for one time

step. However, notice that in some texts $d(y(t), h)/h$ is defined to be the local truncation error.

Recalling Taylor's Theorem in two variables,⁷ we obtain

$$\begin{aligned}\Phi(t, y(t), h) &= f\left(t + \frac{h}{2}, y + \frac{h}{2}f(t, y)\right) \\ &= f(t, y) + \frac{h}{2} \frac{\partial f(t, y)}{\partial t} + \frac{h}{2} f(t, y) \frac{\partial f(t, y)}{\partial y} + \mathcal{O}(h^2),\end{aligned}$$

provided that f , f_t , f_y , f_{tt} , f_{yy} , and f_{ty} are continuous and bounded for $a \leq t \leq b$. Hence,

$$\begin{aligned}\Phi(t, y(t), h) &= f(t, y) + \frac{h}{2}[f_t + f f_y](t, y) + \mathcal{O}(h^2) \\ &= f(t, y) + \frac{h}{2} \frac{d}{dt} f(t, y) + \mathcal{O}(h^2) \\ &= y'(t) + \frac{h}{2} y''(t) + \mathcal{O}(h^2).\end{aligned}$$

Substituting this result into (7.37) and assuming that $y'''(t)$ is bounded and continuous for $a \leq t \leq b$, we have

$$\begin{aligned}d(y(t), h) &= y(t) + h y'(t) + \frac{h^2}{2} y''(t) + \mathcal{O}(h^3) \\ &\quad - y(t) - h y'(t) - \frac{h^2}{2} y''(t) + \mathcal{O}(h^3) = \mathcal{O}(h^3)\end{aligned}\tag{7.38}$$

Hence, if $y'''(t)$ is continuous for $a \leq t \leq b$, we see that the midpoint method has order $p = 2$.

REMARK 7.6 For a system ($n > 1$), it can be verified by Taylor series expansions of each component that the midpoint method has order 2. $\square$

7.3.3 An Error Bound for Single-Step Methods

We now derive an *error bound* for single-step methods (7.25).

THEOREM 7.4

Let $y(t)$ be the solution of (7.1) and let $\{y_j\}_{j=0}^N$ be the numerical solution generated by method (7.25). Furthermore, suppose that

- (a) $\Phi(t, y, h)$ of (7.25) is a continuous function of its arguments;

⁷We review multivariate Taylor polynomials on page 441.

(b) there exists a positive constant M such that

$$\|\Phi(t, y, h) - \Phi(t, \tilde{y}, h)\| \leq M\|y - \tilde{y}\|, \quad (7.39)$$

for all $y, \tilde{y} \in \mathbb{R}^n$, $h \in [0, h_0]$, $t \in [a, b]$ (That is, Φ is Lipschitz in y with Lipschitz constant⁸ M .);

(c) the method (7.25) is consistent and has order $p > 0$; specifically let

$$d(t, y(t), h) = y(t+h) - y(t) - h\Phi(t, y(t), h),$$

and suppose there is a constant D , independent of h , such that

$$\|d(t, y(t), h)\| \leq Dh^{p+1} \quad (7.40)$$

for $t \in [a, b]$, $h \in [0, h_0]$, where D may depend on y and $f(t, y)$.

We then have the following error bound:

$$\max_k \|y_k - y(t_k)\| \leq ch^p \quad (7.41)$$

for $0 \leq h \leq h_0$, where $c = D(e^{M(b-a)} - 1)/M$.

PROOF By (7.25), we have

$$y_{k+1} = y_k + h\Phi(t_k, y_k, h). \quad (7.42)$$

We also obtain

$$y(t_{k+1}) = y(t_k) + h\Phi(t_k, y(t_k), h) + d(t_k, y(t_k), h), \quad (7.43)$$

from the definition of $d(t, y(t), h)$. Hence, letting $e_k = y_k - y(t_k)$, $0 \leq k \leq N-1$, (7.42) and (7.43) imply

$$e_{k+1} = e_k + h[\Phi(t_k, y_k, h) - \Phi(t_k, y(t_k), h)] - d(t_k, y(t_k), h). \quad (7.44)$$

This, with (7.39) and (7.40), in turn implies

$$\|e_{k+1}\| \leq \|e_k\|(1 + hM) + Dh^{p+1}. \quad (7.45)$$

We now apply Lemma 7.1 as follows:

$$d_{n+1} \leq d_n(1 + \delta) + K^*$$

implies

$$d_n \leq d_0 e^{n\delta} + \frac{K^*}{\delta}(e^{n\delta} - 1) \quad \text{with} \quad d_k = \|e_k\|, \quad \delta = hM, \quad K^* = Dh^{p+1}.$$

⁸Here, the norm can be any norm, although the exact value of M depends on the norm.

Now, noting $e_0 = y_0 - y(0) = 0$, we obtain

$$\|e_k\| \leq \frac{D}{M}(e^{M(b-a)} - 1)h^p \quad (7.46)$$

for $0 \leq k \leq N$, which yields (7.41). $\square$

REMARK 7.7 Note that $d(t, y(t), h) = \mathcal{O}(h^{p+1})$ and $\|e_k\| = \mathcal{O}(h^p)$, i.e., the order of the “local discrepancy” or “local truncation error” of the method is one order higher than the error bound. $\square$

REMARK 7.8 To achieve bound (7.41), sufficient smoothness assumptions have been imposed on $y(t)$, the solution of (7.1). If $y(t)$ is not sufficiently smooth, then the method will not yield the high accuracy predicted by (7.41). (Smoothness is used to show (7.40); for example, see the derivation of (7.41) for Euler’s Method on page 391 and the derivation of (7.41) for the midpoint method on page 391.) $\square$

REMARK 7.9 As an easy example, let us verify that conditions (a), (b), and (c) are fulfilled for the midpoint method (7.31), for which

$$\Phi(t_j, y_j, h) = f(t_j + \frac{h}{2}, y_j + \frac{h}{2}f(t_j, y_j))$$

(a) Assuming that conditions of existence-uniqueness Theorem 7.2 hold, condition (a) follows from these conditions.

(b) Furthermore, we have

$$\begin{aligned} \|\Phi(t, y, h) - \Phi(t, \tilde{y}, h)\| &= \|f(t + \frac{h}{2}, y + \frac{h}{2}f(t, y)) \\ &\quad - f(t + \frac{h}{2}, \tilde{y} + \frac{h}{2}f(t, \tilde{y}))\| \\ &\leq L\|y + \frac{h}{2}f(t, y) - \tilde{y} - \frac{h}{2}f(t, \tilde{y})\| \\ &\leq L\|y - \tilde{y}\| + \frac{h}{2}L^2\|y - \tilde{y}\| \leq M\|y - \tilde{y}\|, \end{aligned}$$

where $M = L + L^2h_0/2$.

(c) Finally, we previously verified that (7.40) holds with $p = 2$; hence, the error in the midpoint method is $\mathcal{O}(h^2)$. $\square$

7.3.4 Higher-Order Methods

We now construct some single-step methods of high-order. Two types of single-step methods are Taylor series methods and Runge–Kutta methods.

7.3.4.1 Taylor Series Methods

Consider first *Taylor series methods*, which form one class of single-step methods. These methods are easy to derive. In the past, these methods were seldom used in practice since they required evaluations of high-order derivatives. However, with efficient implementations of automatic differentiation,⁹ these methods are increasingly solving important real-world problems. For example, very high-order Taylor methods (of order 30 or higher) are used, with the aid of automatic differentiation, in the “COSY Infinity” package, which is used world-wide to model atomic particle accelerator beams. (See, for example, [13].)

If $y(t)$, the solution of (7.1), is sufficiently smooth, we see that

$$y(t_{k+1}) = y(t_k) + hy'(t_k) + \frac{h^2}{2}y''(t_k) + \cdots + \frac{h^p}{p!}y^{(p)}(t_k) + \mathcal{O}(h^{p+1}) \quad (7.47)$$

where, using (7.1), these derivatives can be computed explicitly with the multivariate chain rule of the usual calculus. Thus, (7.47) leads to the following numerical scheme:

$$\begin{cases} y_0 = y(a) \\ y_{k+1} = y_k + hf(t_k, y_k) + \frac{h^2}{2} \frac{d}{dt} f(t_k, y_k) + \cdots + \frac{h^p}{p!} \frac{d^{p-1}}{dt^{p-1}} f(t_k, y_k) \end{cases} \quad (7.48)$$

for $k = 0, 1, 2, \dots, N-1$. This is called a Taylor series method. (Note that Euler’s method is a Taylor series method of order $p = 1$.)

By construction, the order of the method (7.48) is p . In weighing the practicality of this method, one should consider the structure of the problem itself, along with the ease (or lack thereof) of computing the derivatives. For example, with $n = 1$, we must compute

$$\begin{aligned} \frac{d}{dt} f(t, y) &= \frac{\partial f}{\partial t} + f \frac{\partial f}{\partial y}, \\ \frac{d^2}{dt^2} f(t, y) &= \frac{\partial^2 f}{\partial t^2} + 2f \frac{\partial^2 f}{\partial t \partial y} + f^2 \frac{\partial^2 f}{\partial y^2} + \frac{\partial f}{\partial t} \frac{\partial f}{\partial y} + f \left(\frac{\partial f}{\partial y} \right)^2, \\ &\text{etc.} \end{aligned}$$

If f is mildly complicated, then it is impractical to compute these formulas by hand;¹⁰ also, observe that, for $n > 1$, the number of terms can become

⁹These implementations can be very sophisticated

¹⁰but this does not rule out automatic differentiation

large, although many may be zero; thus, an implementation of automatic differentiation should take advantage of the structure in f .

Example 7.3

Consider

$$\begin{cases} y'(t) = f(t, y) = t + y, \\ y(1) = 2, \end{cases}$$

which has exact solution $y(t) = -t - 1 + 4e^{-1}e^t$. The Taylor series method of order 2 for this example has

$$f(t, y) = t + y$$

and

$$\frac{d}{dt}f(t, y) = \frac{\partial f}{\partial t} + f \frac{\partial f}{\partial y} = 1 + t + y.$$

Therefore,

$$\begin{aligned} y_{k+1} &= y_k + hf(t_k, y_k) + \frac{h^2}{2} \frac{d}{dt}f(t_k, y_k) \\ &= y_k + h(t_k + y_k) + \frac{h^2}{2}(1 + t_k + y_k). \end{aligned}$$

Letting $h = 0.1$, we obtain the following results:

k	t_k	y_k (Euler)	y_k (T.S. order 2)	$y(t_k)$ (Exact)
0	1	2	2	2
1	1.1	2.3	2.32	2.3207
2	1.2	2.64	2.6841	2.6856
3	1.3	3.024	3.0969	3.0994
4	1.4	3.4564	3.5636	3.5673

□

7.3.4.2 Runge–Kutta Methods

We now consider single-step Runge–Kutta methods, whose associated function $\Phi(t, y, h)$ requires (possibly repeated) function evaluations of $f(t, y)$ but *not* its derivatives. (Two examples are Euler's Method and Midpoint Method.) In general, single-step Runge–Kutta methods have the form:

$$\begin{cases} y_0 = y(a) \\ y_{k+1} = y_k + h\Phi(t_k, y_k, h) \end{cases} \quad (7.49)$$

where

$$\begin{aligned} \Phi(t, y, h) &= \sum_{r=1}^R c_r K_r, \\ K_1 &= f(t, y), \\ K_r &= f\left(t + a_r h, y + h \sum_{s=1}^{r-1} b_{rs} K_s\right) \end{aligned}$$

and

$$a_r = \sum_{s=1}^{r-1} b_{rs}, \quad r = 2, 3, \dots, R.$$

Such a method is called an *R-stage Runge-Kutta method*. Notice that Euler's method is a one-stage Runge-Kutta method and the midpoint method is a two-stage Runge-Kutta method with $c_1 = 0$, $c_2 = 1$, $a_2 = \frac{1}{2}$, $b_{21} = \frac{1}{2}$, i.e.,

$$y_{k+1} = y_k + hf \left(t_k + \frac{h}{2}, y_k + \frac{h}{2} f(t_k, y_k) \right).$$

We now consider in detail the general two-stage Runge-Kutta method, i.e.,

$$y_{k+1} = y_k + hc_1 f(t_k, y_k) + hc_2 f(t_k + a_2 h, y_k + a_2 h f(t_k, y_k)). \quad (7.50)$$

Let's see if there are other Runge-Kutta methods (2-stage) of order $p = 2$ besides the midpoint method. For simplicity, let $n = 1$. By (7.32), we need to consider

$$\begin{aligned} & y(t+h) - [y(t) + hc_1 f(t, y) + hc_2 f(t + a_2 h, y + a_2 h f(t, y))] \\ &= y(t+h) - y(t) - hc_1 f(t, y) \\ &\quad - hc_2 \left[f(t, y) + \frac{\partial f(t, y)}{\partial t} a_2 h + \frac{\partial f(t, y)}{\partial y} a_2 h f(t, y) \right] + \mathcal{O}(h^3) \\ &= y(t) + hf(t, y) + \frac{h^2}{2} \left[\frac{\partial f(t, y)}{\partial t} + \frac{\partial f(t, y)}{\partial y} f(t, y) \right] - y(t) \\ &\quad - h(c_1 + c_2) f(t, y) - h^2 c_2 a_2 \frac{\partial f(t, y)}{\partial t} - h^2 c_2 a_2 \frac{\partial f(t, y)}{\partial y} f(t, y) + \mathcal{O}(h^3) \\ &= \mathcal{O}(h^3), \end{aligned}$$

if $c_1 + c_2 = 1$ and $c_2 a_2 = 1/2$. Letting $c_2 = \alpha$, then $c_1 = 1 - \alpha$, $a_2 = 1/(2\alpha)$, and we obtain a family of order $p = 2$ schemes, i.e.,

$$\begin{cases} y_0 = y(t_0) \\ y_{k+1} = y_k + h\Phi(t_k, y_k, h), \end{cases} \quad (7.51)$$

where

$$\Phi(t, y, h) = (1 - \alpha) f(t, y) + \alpha f \left(t + \frac{h}{2\alpha}, y + \frac{h}{2\alpha} f(t, y) \right).$$

When $\alpha = 1$, this gives the midpoint method and when $\alpha = 1/2$, this gives what is called the improved or modified Euler method.

An analysis analogous to the second-order case can be performed for methods of order $p = 3$ and order $p = 4$. The most well-known Runge-Kutta

scheme (from elementary numerical analysis texts) is 4-th order; it has the form:

$$\begin{aligned} y_0 &= y(t_0) \\ y_{k+1} &= y_k + \frac{h}{6}[K_1 + 2K_2 + 2K_3 + K_4] \\ K_1 &= f(t_k, y_k) \\ K_2 &= f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2}K_1\right) \\ K_3 &= f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2}K_2\right) \\ K_4 &= f(t_k + h, y_k + hK_3), \end{aligned} \tag{7.52}$$

i.e.,

$$\Phi(t_k, y_k, h) = \frac{h}{6}[K_1 + 2K_2 + 2K_3 + K_4].$$

Notice that in single-step methods, $y_{k+1} = y_k + h\Phi(t_k, y_k, h)$, $h\Phi(t_k, y_k, h)$ is an approximation to the “rise” in y in going from t_k to $t_k + h$. In the fourth-order Runge–Kutta method, $\Phi(t_k, y_k, h)$ is a weighted average of approximate “slopes” K_1, K_2, K_3, K_4 evaluated at $t_k, t_k + h/2, t_k + h/2$ and $t_k + h$, respectively.

Example 7.4

Consider $y'(t) = t + y, y(1) = 2$, with $h = 0.1$. We obtain

k	t_k	Euler	Runge–Kutta order 2 (Modified Euler)	Runge–Kutta order 4	$y(t_k)$ (exact)
0	1	2	2	2	2
1	1.1	2.30	2.32	2.32068	2.32068
2	1.2	2.64	2.6841	2.68561	2.68561
3	1.3	3.024	3.09693	3.09943	3.09944

□

Higher-order Runge–Kutta methods are sometimes used, such as in the Runge–Kutta–Fehlberg method we introduce in section 7.4 below.

7.3.4.3 Stability of Runge–Kutta Methods

We now consider *stability* of Runge–Kutta methods. Suppose that at the k -th step, due to rounding errors, we actually don’t obtain y_k , but we obtain z_k such that $y_k - z_k \neq 0$. The error at the k -th step will influence subsequent values, so that at the N -th and final step, we have $y_N - z_N \neq 0$, where y_N is what would be obtained if we applied exact computations (without roundoff error) to the iteration equation (7.25), and z_N is the actual computed result, assuming an error was made at the k -th step.

DEFINITION 7.4 We say that the Runge–Kutta method is numerically stable if there is a constant c independent of h such that

$$\|y_N - z_N\| \leq c\|y_k - z_k\| \quad \text{for all } k \leq N, \quad (7.53)$$

where $h = (b - a)/N$.

Suppose that conditions (a) and (b) of Theorem 7.4 (our error bound theorem, on page 393) are satisfied. Then it is straightforward to show that the method is stable. Considering $j \geq k$, we have

$$z_{j+1} = z_j + h\Phi(t_j, z_j, h). \quad (7.54)$$

Hence,

$$z_{j+1} - y_{j+1} = z_j - y_j + h[\Phi(t_j, z_j, h) - \Phi(t_j, y_j, h)] \quad (7.55)$$

for $k \leq j \leq N - 1$ and thus,

$$\|z_{j+1} - y_{j+1}\| \leq (1 + hM)\|z_j - y_j\|, \quad k \leq j \leq N - 1,$$

from which we see that

$$\begin{aligned} \|z_N - y_N\| &\leq (1 + hM)^{N-k}\|z_k - y_k\| \\ &\leq (1 + hM)^N\|z_k - y_k\| \leq e^{hMN}\|z_k - y_k\| = e^{M(b-a)}\|z_k - y_k\| \\ &= c\|z_k - y_k\|. \end{aligned}$$

Hence, Runge–Kutta methods, under the stated conditions, are stable in the sense that (7.53) is satisfied. This implies that an error $\|z_k - y_k\|$ will not be magnified by more than a constant c at final time t_N , i.e., “small errors” have “small effect.”

The above definition of stability is not satisfactory if the constant c is very large. Consider, for example, Euler’s method applied to the scalar equation $y' = \lambda y$, $\lambda = \text{constant}$. Then Euler’s scheme gives $y_{j+1} = y_j(1 + \lambda h)$, $0 \leq j \leq N - 1$. An error, say at $t = t_k$, will cause us to compute $z_{j+1} = z_j(1 + \lambda h)$ instead and hence $|z_{j+1} - y_{j+1}| = |1 + \lambda h||z_j - y_j|$, $k \leq j \leq N - 1$. Thus, the error will be *magnified* if $|1 + \lambda h| > 1$, will remain the same if $|1 + \lambda h| = 1$, and will be *suppressed* if $|1 + \lambda h| < 1$. Consider the problem

$$y' = -1000y + 1000t^2 + 2t, \quad 0 \leq t \leq 1, \quad y(0) = 0,$$

whose exact solution is $y(t) = t^2$, $0 \leq t \leq 1$. We find for Euler’s method that $|z_{j+1} - y_{j+1}| = |1 - 1000h||z_j - y_j|$, $0 \leq j \leq N - 1$. The error will be suppressed if $|1 - 1000h| < 1$, i.e., $0 \leq h \leq 0.002$. Consider the following table:

h	N	y_N
1	1	0
0.1	10	9×10^{16}
0.01	10^2	overflow
0.001	10^3	0.99999900
0.0001	10^4	0.99999990
0.00001	10^5	0.99999999

For $h > .002$, small errors are violently magnified. For example, for $h = .01$, the errors are magnified by $|1 - \frac{1000}{100}| = 9$ at each time step. However, note that the error bound given by Theorem 7.3, that is,

$$|y_N - y(1)| \leq \frac{Mh}{2L} |e^{L(b-a)} - 1| = \frac{e^{1000} - 1}{10^5},$$

($M = 2, L = 1000, a = 0, b = 1, h = 0.01$) is still valid, but the error bound is so large that it is practically meaningless.

This discussion motivates a second concept of stability that will be important when we discuss stiff systems.

DEFINITION 7.5 A numerical method for solution of (7.1) is called *absolutely stable* if when applied to the scalar equation $y' = \lambda y, t \geq 0$, it yields values $\{y_j\}_{j \geq 0}$ with the property that $y_j \rightarrow 0$ as $j \rightarrow \infty$. The set of values λh for which a method is absolutely stable is called the *set of absolute stability*.

Example 7.5

(Absolute stability of Euler's method and the midpoint method)

1. Euler's Method applied to $y' = \lambda y$ yields $y_{j+1} = y_j(1 + \lambda h)$, whence

$$y_j = y_0(1 + \lambda h)^{j+1}.$$

Clearly, assuming that λ is real, $y_j \rightarrow 0$ as $j \rightarrow \infty$ if and only if $|1 + \lambda h| < 1$ or $-2 < \lambda h < 0$. Hence, the interval of absolute stability of Euler's method is $(-2, 0)$.

2. The Midpoint Method applied to $y' = \lambda y$ yields

$$y_{j+1} = y_j + h\lambda(y_j + \frac{h}{2}\lambda y_j) = y_j(1 + \lambda h + \frac{\lambda^2 h^2}{2}).$$

Hence, $y_j \rightarrow 0$ as $j \rightarrow \infty$ if $|1 + \lambda h + \lambda^2 h^2/2| < 1$, which for λ real leads to an interval of absolute stability $(-2, 0)$. (Other examples are given in the exercises.)



In general, explicit Runge–Kutta methods require significantly small h for accurate approximations of problems with large $|\lambda|$. (Notice that the linear case with λ models the nonlinear case $y' = f(t, y)$ with Lipschitz constant $L \approx \lambda$.) These methods should not be used for such problems or their system analogs (stiff system). We will consider later methods suitable for such problems.

7.4 Error Control and the Runge–Kutta–Fehlberg Method

Consider $n = 1$. Suppose that $y(t)$ rapidly increases (or decreases) at some value of t , e.g., $t = c$. (See Figure 7.2.) Large time steps may provide high

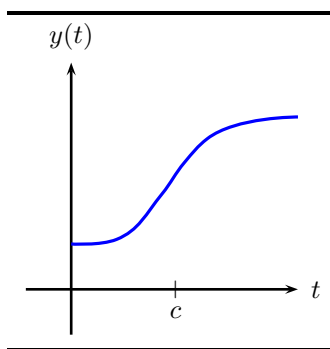


FIGURE 7.2: Sudden increase in a solution.

accuracy in regions where y does not vary rapidly with t but in regions where y varies rapidly, small h may be essential to obtain the desired accuracy. However, the different regions may be unknown beforehand.

It is straightforward to derive single-step methods with variable time steps. For example, with $a = t_0 < t_1 < \cdots < t_{N-1} < t_N = b$, and $h_j = t_{j+1} - t_j$, $0 \leq j \leq N - 1$, Euler's scheme becomes

$$\begin{cases} y_0 = y(t_0) = y(a) \\ y_{j+1} = y_j + h_j f(t_j, y_j). \end{cases}$$

However, we would like our method to “sense” when the time steps $h_j = t_{j+1} - t_j$ should be decreased depending on the behavior of the solution $y(t)$

which is not known beforehand. Therefore, we would like to estimate the error committed by the scheme and reduce the time step if the errors are too large.

Consider the general Runge–Kutta scheme

$$\begin{cases} y_0 = y(t_0) \\ y_{j+1} = y_j + h_j \Phi(t_j, y_j, h_j). \end{cases} \quad (7.56)$$

The quantity $y(t_{j+1}) - y_{j+1}$ is called the *global error* at time step t_j . Unfortunately, $y(t_{j+1}) - y_{j+1}$ is hard to estimate directly so we focus on the *local error*.

To define local error, let $y(t)$, $t_j \leq t \leq t_{j+1}$, be the solution of IVP (7.1) and consider

$$\begin{cases} u'(t_j) = f(t, u(t)), & t_j \leq t \leq t_{j+1} \\ u(t_j) = y_j. \end{cases} \quad (7.57)$$

The local error at t_{j+1} is defined by the quantity $u(t_{j+1}) - y_{j+1}$. The local error is thus a measure of the error committed by the numerical method in just a single step. Figure 7.3 illustrates this for $n = 1$.

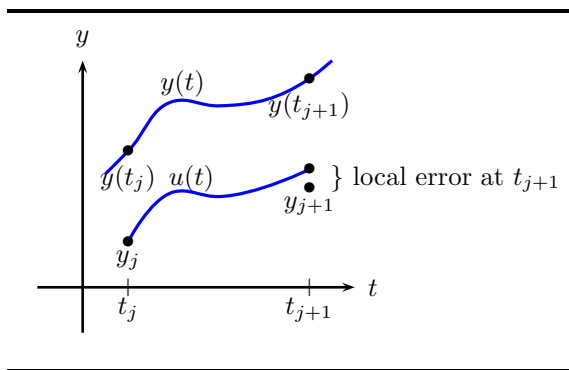


FIGURE 7.3: Local error in integrating an ODE.

Since the global error $= y(t_{j+1}) - y_{j+1} = y(t_{j+1}) - u(t_{j+1}) + u(t_{j+1}) - y_{j+1}$, the global error is the sum of the local error and a quantity $y(t_{j+1}) - u(t_{j+1})$ which measures the “stability” of the ODE $y' = f(t, y)$ in the sense that “small” deviations at $t = t_j$ should produce “small” effects at $t = t_j + h_j$ for h_j small as $y(t)$ and $u(t)$ are solutions of the same ODE with different starting values.

Thus, we concentrate on estimating local errors. First, recall that the single-step method of order p has local truncation error

$$d(t, y(t), h) = y(t + h) - y(t) - h\Phi(t, y(t), h) = \mathcal{O}(h^{p+1}), \quad (7.58)$$

where $y(t)$ satisfies the ODE.

In particular, $u(t)$ satisfies (by (7.57)) the ODE so

$$d(t_j, u(t_j), h_j) = u(t_{j+1}) - u(t_j) - h_j \Phi(t_j, u(t_j), h_j) = \mathcal{O}(h_j^{p+1}). \quad (7.59)$$

In addition, $0 = y_{j+1} - y_j - h_j \Phi(t_j, y_j, h_j)$ so

$$u(t_{j+1}) - y_{j+1} = \mathcal{O}(h^{p+1}) \quad (7.60)$$

since $u(t_j) = y_j$. Thus, the local error of a method of order p is $\mathcal{O}(h^{p+1})$.

Suppose now we have a second Runge–Kutta method of the form (7.49). Specifically, we assume that with the initial starting value y_j we compute

$$\tilde{y}_{j+1} = y_j + h_j \tilde{\Phi}(t_j, y_j, h_j) \quad (7.61)$$

by a method of order $q \geq p + 1$ (so presumably $\tilde{y}_{j+1}$ is more accurate than y_{j+1}). For this order q method, we have that

$$\tilde{d}(t_j, u(t_j), h_j) = u(t_{j+1}) - u(t_j) - h_j \tilde{\Phi}(t_j, u(t_j), h_j) = \mathcal{O}(h_j^{q+1}).$$

But since $u(t_j) = y_j$, combining this with (7.61) gives

$$u(t_{j+1}) - y_{j+1} = \mathcal{O}(h_j^{q+1}) + \tilde{y}_{j+1} - y_{j+1}. \quad (7.62)$$

Therefore, by (7.62), the local error can be estimated to $\mathcal{O}(h^{q+1})$ by $\tilde{y}_{j+1} - y_{j+1}$.

Hence, by computing $\tilde{y}_{j+1}$ and y_{j+1} at each time step we can estimate the local error and adjust the time step accordingly. One way of doing this is as follows. We assume that $\epsilon > 0$ is a given required tolerance and insist at each time step that

$$\|\tilde{y}_{j+1} - y_{j+1}\| \leq \epsilon(t_{j+1} - t_j) \quad (7.63)$$

If (7.63) is not satisfied, the time step is reduced, $\tilde{y}_{j+1}$ and y_{j+1} are recomputed, and the local error again estimated, etc. It can be shown (exercise) that if (7.63) holds and $f(t, y)$ and $\Phi(t, y, h)$ are Lipschitz continuous with respect to y then the global error satisfies

$$\|y(t_j) - y_j\| \leq c\epsilon \quad (7.64)$$

for some constant c independent of ϵ and h . Thus, this time step strategy controls the global error in the sense of (7.64).

A popular error control method is the Runge–Kutta–Fehlberg Method. In the RKF method, we choose a pair ($p = 4, q = 5$) of Runge–Kutta methods that together only require six function evaluations per time step. These are based on the following function evaluations:

$$\begin{aligned}
K_1 &= f(t_j, y_j), \\
K_2 &= f\left(t_j + \frac{h}{4}, y_j + \frac{h}{4}K_1\right), \\
K_3 &= f\left(t_j + \frac{3h}{8}, y_j + h\left(\frac{3}{32}K_1 + \frac{9}{32}K_2\right)\right), \\
K_4 &= f\left(t_j + \frac{12h}{13}, y_j + h\left(\frac{1932}{2197}K_1 - \frac{7200}{2197}K_2 + \frac{7296}{2197}K_3\right)\right), \\
K_5 &= f\left(t_j + h, y_j + h\left(\frac{439}{216}K_1 - 8K_2 + \frac{3680}{513}K_3 - \frac{845}{4104}K_4\right)\right), \\
K_6 &= f\left(t_j + \frac{h}{2}, y_j + h\left(-\frac{8}{27}K_1 + 2K_2 - \frac{3544}{2565}K_3 + \frac{1859}{4104}K_4 - \frac{11}{40}K_5\right)\right).
\end{aligned}$$

Then the 4-*th*-order method is:

$$y_{j+1} = y_j + h\left(\frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4104}K_4 - \frac{1}{5}K_5\right) \quad (7.65)$$

and the 5-*th*-order method is:

$$\tilde{y}_{j+1} = y_j + h\left(\frac{16}{135}K_1 + \frac{6656}{12825}K_3 + \frac{28561}{56430}K_4 - \frac{9}{50}K_5 + \frac{2}{55}K_6\right) \quad (7.66)$$

and the local error $\approx \tilde{y}_{j+1} - y_{j+1}$.

A possible procedure for implementing the RKF method has the form:

1. h given, ϵ given.
2. $t_{j+1} = t_j + h$
3. calculate $y_{j+1}, \tilde{y}_{j+1}$
4. if $\|y_{j+1} - \tilde{y}_{j+1}\| \leq \epsilon(t_{j+1} - t_j)$, then output y_{j+1}, t_{j+1} and let $j = j + 1$
5. let $q = \left(\frac{\frac{1}{2}\epsilon h}{\|y_{j+1} - \tilde{y}_{j+1}\|}\right)^{\frac{1}{4}}$
6.
 - if $q \leq 0.1$, let $h = 0.1h$
 - if $0.1 \leq q \leq 4$, let $h = qh$
 - if $q > 4$, let $h = 4h$
 - if $h > h_{max}$, let $h = h_{max}$
7. go to (2) until $t_{j+1} > t_{max}$

Step (5) results from the following argument. We wish to determine qh so that using step size qh ,

$$\epsilon > \frac{\|y_{j+1}^{(qh)} - \tilde{y}_{j+1}^{(qh)}\|}{qh} = \mathcal{O}((qh)^4) \approx kq^4h^4 = q^4h^4k \approx \frac{q^4}{h}\|y_{j+1} - \tilde{y}_{j+1}\|$$

$$\text{Hence, } q^4 < \frac{h\epsilon}{\|y_{j+1} - \tilde{y}_{j+1}\|}.$$

Incorporating a “safety factor” of $\frac{1}{2}$, $q < (\frac{\frac{1}{2}h\epsilon}{\|y_{j+1} - \tilde{y}_{j+1}\|})^{\frac{1}{4}}$. Note that error control methods can be applied to other numerical methods for solving (7.1) such as extrapolation methods.

7.5 Multistep Methods

As an introduction to multistep methods, consider a simple example. Suppose that $n = 1$ and we use Simpson’s method to approximate

$$\int_{t_j}^{t_{j+2}} y'(t)dt = \int_{t_j}^{t_{j+2}} f(t, y(t))dt,$$

where $y(t)$ solves (7.1). Then

$$y(t_{j+2}) - y(t_j) \approx \frac{2h}{6}[f(t_{j+2}, y(t_{j+2})) + 4f(t_{j+1}, y(t_{j+1})) + f(t_j, y(t_j))]$$

from which the 2-step Simpson’s method follows:

$$y_{j+2} = y_j + \frac{h}{3}[f(t_{j+2}, y_{j+2}) + 4f(t_{j+1}, y_{j+1}) + f(t_j, y_j)]. \quad (7.67)$$

In addition to requiring two starting values, y_0 and y_1 for Simpson’s method, y_{j+2} is given *implicitly*, since y_{j+2} occurs on the right-hand side of (7.67). Thus, generally, a nonlinear system has to be solved at each step.

DEFINITION 7.6 A k -step multistep method for solution of (7.1) is a method of the form

$$\begin{cases} y_0, y_1, \dots, y_{k-1} \text{ given starting values} \\ \alpha_k y_{j+k} + \alpha_{k-1} y_{j+k-1} + \dots + \alpha_0 y_j \\ \quad = h(\beta_k f_{j+k} + \beta_{k-1} f_{j+k-1} + \dots + \beta_0 f_j) \end{cases} \quad (7.68)$$

for $j = 0, 1, 2, \dots, N - k$ or

$$\sum_{l=0}^k \alpha_l y_{j+l} = h \sum_{l=0}^k \beta_l f_{j+l}$$

for $j = 0, 1, 2, \dots, N - k$, where $\{\alpha_l\}_{l=0}^k, \{\beta_l\}_{l=0}^k$ are given constants, independent of t or h , and $f_{j+l} = f(t_{j+l}, y_{j+l})$.

We assume for definiteness that $\alpha_k = 1$ and $|\alpha_0| + |\beta_0| > 0$ so that we do indeed have a k -step method. Note that to start such a method we need values $\{y_j\}_{j=0}^{k-1}$, which are generally obtained using $y_0 = y(t_0)$ and the use of a single-step Runge–Kutta method for $k - 1$ steps. (A Runge–Kutta method of the same (or higher) order accuracy as the multistep method would be used.)

DEFINITION 7.7 If $\beta_k = 0$ the method is called *explicit*; if $\beta_k \neq 0$ the method is called *implicit*, and requires solution of a nonlinear system to compute y_{j+k} for each step.

Example 7.6

Euler's method is a single-step explicit method ($k = 1$) with $\alpha_1 = 1, \alpha_0 = -1, \beta_1 = 0, \beta_0 = 1$. Simpson's method is a two-step implicit method ($k = 2$) with $\alpha_2 = 1, \alpha_1 = 0, \alpha_0 = -1, \beta_2 = 1/3, \beta_1 = 4/3, \beta_0 = 1/3$. $\square$

A frequently used one-step implicit method called the trapezoidal rule is given by

$$y_{j+1} - y_j = h \left(\frac{1}{2} f_{j+1} + \frac{1}{2} f_j \right), \quad (7.69)$$

which is, of course, obtained by approximating the integral in

$$y(t_{j+1}) - y(t_j) = \int_{t_j}^{t_{j+1}} f(t, y(t)) dt$$

by the trapezoidal rule.

There are many ways to derive multistep methods of the form (7.68), e.g., by Taylor series, approximate integration, polynomial approximation, etc. A class of k -step methods of the form

$$y_{j+k} - y_{j+k-1} = h \sum_{l=0}^k \beta_l f_{j+l}$$

is the class of *Adams methods*. In particular, *Adams–Bashforth methods* have $\beta_k = 0$ and *Adams–Moulton methods* have $\beta_k \neq 0$. A classical work in which the derivation of these methods is clearly explained is [84].

We now consider *convergence, stability, and consistency* of multistep methods.

DEFINITION 7.8 The k -step method (7.68) is convergent to solution $y(t)$ of (7.1) if

$$\lim_{\substack{j \rightarrow \infty, h \rightarrow 0 \\ jh = t-a}} y_j = y(t) \quad (7.70)$$

for all $t \in [a, b]$, provided that $\lim_{h \rightarrow 0} y_j = y_0$ for $0 \leq j \leq k-1$.

Associated with the multistep method (7.68), we define a linear operator $\mathcal{L}$ (for $n = 1$) by

$$\mathcal{L}[y(t); h] = \sum_{l=0}^k [\alpha_l y(t+lh) - h\beta_l y'(t+lh)]. \quad (7.71)$$

Expanding the right-hand side in a Taylor series about t , we have

$$\mathcal{L}[y(t); h] = c_0 y(t) + c_1 h y'(t) + c_2 h^2 y''(t) + \cdots + c_q h^q y^{(q)}(t) + \cdots, \quad (7.72)$$

where $c_0, c_1, c_2, \dots, c_q$ are constants depending on the coefficients α_j, β_j , $0 \leq j \leq k$. These constants are given explicitly by

$$\begin{cases} c_0 = \sum_{l=0}^k \alpha_l, \\ c_1 = \sum_{l=1}^k l\alpha_l - \sum_{l=0}^k \beta_l, \\ c_q = \frac{1}{q!}(\alpha_1 + 2^q \alpha_2 + \cdots + k^q \alpha_k) - \frac{1}{(q-1)!}(\beta_1 + 2^{q-1} \beta_2 + \cdots + k^{q-1} \beta_k), \end{cases} \quad (7.73)$$

for $q = 2, 3, \dots$.

With this, we give an alternate definition of order of accuracy, for multistep methods.

DEFINITION 7.9 *The k -step method (7.68) is said to be of order of accuracy p if in (7.73) $c_0 = c_1 = c_2 = \cdots = c_p = 0$ but $c_{p+1} \neq 0$.*

To see why this definition is reasonable, consider the local error for multistep methods. For simplicity, consider the general two-step method and $n = 1$:

$$\alpha_2 y_{j+2} = -\alpha_1 y_{j+1} - \alpha_0 y_j + h(\beta_2 f(t_{j+2}, y_{j+2}) + \beta_1 f(t_{j+1}, y_{j+1}) + \beta_0 f(t_j, y_j)).$$

Let $y_{j+1} = y(t_{j+1})$ and $y_j = y(t_j)$, that is y_j and y_{j+1} are exact. Let

$$\begin{aligned} \alpha_2 \tilde{y}_{j+2} &= -\alpha_1 y(t_{j+1}) - \alpha_0 y(t_j) \\ &\quad + h(\beta_2 f(t_{j+2}, \tilde{y}_{j+2}) + \beta_1 f(t_{j+1}, y(t_{j+1})) + \beta_0 f(t_j, y(t_j))), \end{aligned}$$

so the local error is

$$d_j = |y(t_{j+2}) - \tilde{y}_{j+2}|.$$

By reasoning similar to that used for one-step methods, if $d_j = \mathcal{O}(h^{p+1})$, then the global error should be $\mathcal{O}(h^p)$. Therefore, consider

$$\begin{aligned}
 d_j &= |y(t_{j+2}) - \tilde{y}_{j+2}| \\
 &= \frac{1}{|\alpha_2|} |\alpha_2 y(t_{j+2}) + \alpha_1 y(t_{j+1}) + \alpha_0 y(t_j) - h\beta_2 f(t_{j+2}, \tilde{y}_{j+2}) \\
 &\quad - h\beta_1 f(t_{j+1}, y(t_{j+1})) - h\beta_0 f(t_j, y(t_j))| \\
 &\leq \frac{1}{|\alpha_2|} |\alpha_2 y(t_{j+2}) + \alpha_1 y(t_{j+1}) + \alpha_0 y(t_j) - h\beta_2 f(t_{j+2}, y(t_{j+2})) \\
 &\quad - h\beta_1 f(t_{j+1}, y(t_{j+1})) - h\beta_0 f(t_j, y(t_j))| \\
 &\quad + \frac{1}{|\alpha_2|} |h\beta_2| |f(t_{j+2}, y(t_{j+2})) - f(t_{j+2}, \tilde{y}_{j+2})|.
 \end{aligned}$$

Hence,

$$\begin{aligned}
 d_j &\leq \frac{1}{|\alpha_2|} |\alpha_2 y(t_{j+2}) + \alpha_1 y(t_{j+1}) + \alpha_0 y(t_j) - h\beta_2 y'(t_{j+2}) \\
 &\quad - h\beta_1 y'(t_{j+1}) - h\beta_0 y'(t_j)| + hL \left| \frac{\beta_2}{\alpha_2} \right| d_j
 \end{aligned}$$

assuming that f satisfies a Lipschitz condition. Thus,

$$\begin{aligned}
 (1 - hL \left| \frac{\beta_2}{\alpha_2} \right|) d_j &\leq \frac{1}{|\alpha_2|} |\alpha_2 y(t_{j+2}) + \alpha_1 y(t_{j+1}) + \alpha_0 y(t_j) - h\beta_2 y'(t_{j+2}) \\
 &\quad - h\beta_1 y'(t_{j+1}) - h\beta_0 y'(t_j)|
 \end{aligned}$$

(We assume that $1 - h_0 L \left| \frac{\beta_2}{\alpha_2} \right| > 0$ where $0 \leq h \leq h_0$.) Expanding in Taylor series,

$$\begin{aligned}
 \left(1 - hL \left| \frac{\beta_2}{\alpha_2} \right| \right) d_j &\leq \frac{1}{|\alpha_2|} \alpha_2 \cdot \\
 &\quad \left[y(t_j) + y'(t_j)2h + y''(t_j) \frac{(2h)^2}{2} + y'''(t_j) \frac{(2h)^3}{3!} + \dots \right] \\
 &\quad + \alpha_1 [y(t_j) + y'(t_j)h + y''(t_j) \frac{h^2}{2} + \dots] + \alpha_0 y(t_j) \\
 &\quad - h\beta_2 [y'(t_j) + y''(t_j)2h + y'''(t_j) \frac{(2h)^2}{2} + \dots] \\
 &\quad - h\beta_1 [y'(t_j) + y''(t_j)h + y'''(t_j) \frac{h^2}{2} + \dots] - h\beta_0 y'(t_j) \\
 &= (\alpha_2 + \alpha_1 + \alpha_0)y(t_j) + (2\alpha_2 + \alpha_1 - \beta_2 - \beta_1 - \beta_0)hy'(t_j) \\
 &\quad + (2\alpha_2 + \frac{1}{2}\alpha_1 - 2\beta_2 - \beta_1)h^2y''(t_j) \\
 &\quad + (\frac{8}{6}\alpha_2 - \frac{1}{6}\alpha_1 - 2\beta_2 - \frac{1}{2}\beta_1)h^3y'''(t_j) + \dots.
 \end{aligned}$$

But the right-hand side of the above inequality is exactly

$$c_0 y(t_j) + c_1 h y'(t_j) + c_2 h^2 y''(t_j) + c_3 h^3 y'''(t_j) + \cdots$$

where $c_i, i = 0, 1, 2, \dots$ satisfy (7.73). Specifically,

$$\begin{aligned} c_0 &= \alpha_2 + \alpha_1 + \alpha_0 \\ c_1 &= 2\alpha_2 + \alpha_1 - \beta_2 - \beta_1 - \beta_0 \\ c_2 &= \frac{1}{2}(4\alpha_2 + \alpha_1) - (2\beta_2 + \beta_1) \\ c_3 &= \frac{1}{6}(8\alpha_2 + \alpha_1) - \frac{1}{2}(4\beta_2 + \beta_1) \\ &\vdots \end{aligned}$$

In fact, by examining our argument, the local error satisfies

$$d_j \leq \frac{1}{|\alpha_2|} |\mathcal{L}[y(t_j), h]| + \frac{h}{|\alpha_2|} |\beta_2| L d_j.$$

In general,

$$cd_j \leq \frac{1}{|\alpha_k|} |\mathcal{L}[y(t_j), h]| \leq \mathcal{O}(h^{p+1})$$

if $c_0 = c_1 = c_2 = \cdots = c_p = 0$ but $c_{p+1} \neq 0$, where $c = 1 - \frac{h_0}{|\alpha_k|} |\beta_k| L > 0$ for $0 \leq h \leq h_0$ and sufficiently small h_0 .

Thus, for sufficient smoothness of the solution $y(t)$, the local error is $\mathcal{O}(h^{p+1})$ which implies that the global error is $\mathcal{O}(h^p)$.

Example 7.7

It is straightforward to see that Euler's method is order $p = 1$, (This agrees with the analysis for one-step methods.) Specifically using (7.68) and (7.73), $\alpha_1 = 1$, $\alpha_0 = -1$, $\beta_1 = 0$ and $\beta_0 = 1$ for Euler's method. Thus, $c_0 = \alpha_1 + \alpha_0 = 0$, $c_1 = \alpha_1 - \beta_1 - \beta_0 = 0$ and $c_2 = \frac{1}{2}\alpha_1 - \frac{1}{1}\beta_1 = \frac{1}{2} \neq 0$. Hence, by Definition 7.9, Euler's method has order of accuracy $p = 1$. $\square$

Consider the one-step implicit Trapezoidal method. For this method, $\alpha_1 = 1$, $\alpha_0 = -1$, $\beta_1 = 1/2$, and $\beta_0 = 1/2$. Hence, $c_0 = \alpha_1 + \alpha_0 = 0$, $c_1 = \alpha_1 - \beta_1 - \beta_0 = 0$, $c_2 = \frac{1}{2}\alpha_1 - \frac{1}{1}\beta_1 = 0$, $c_3 = \frac{1}{6}\alpha_1 - \frac{1}{2}\beta_1 = \frac{1}{6} - \frac{1}{4} = -\frac{1}{12} \neq 0$. Thus, the Trapezoidal method has order of accuracy $p = 2$. In addition, it is straightforward to show that the two-step implicit Simpson's method (defined by Equation (7.67) on page 405) has order $p = 4$ (Exercise 21 on page 435).

We now present an alternative formulation of consistency for k -step methods.

DEFINITION 7.10 *The k -step method is consistent if it has order $p \geq 1$.*

It follows that method (7.68) is consistent if and only if $c_0 = c_1 = 0$ in (7.72) and (7.73), that is if and only if

$$\sum_{l=0}^k \alpha_l = 0 \quad \text{and} \quad \sum_{l=1}^k l\alpha_l = \sum_{l=0}^k \beta_l \quad (7.74)$$

REMARK 7.10 It is straightforward to show that this agrees with our earlier definition of consistency for explicit one-step methods. (Consider $\Phi(t_j, y_j, h) = \beta_0 f(t_j, y_j)$ with $\beta_1 = 0$.) $\square$

In addition, we associate the following polynomials $\rho(z)$ and $\sigma(z)$ of a single complex variable z with multistep methods.

$$\begin{cases} \rho(z) = \alpha_0 + \alpha_1 z + \alpha_2 z^2 + \cdots + \alpha_k z^k = \sum_{l=0}^k \alpha_l z^l. \\ \sigma(z) = \beta_0 + \beta_1 z + \beta_2 z^2 + \cdots + \beta_k z^k = \sum_{l=0}^k \beta_l z^l. \end{cases} \quad (7.75)$$

It follows that the k -step method (7.68) is consistent if and only if

$$\begin{cases} \rho(1) = 0, \\ \rho'(1) = \sigma(1). \end{cases} \quad (7.76)$$

We are now going to define stability, but first we motivate the definition. Consider the scalar initial-value problem $y'(t) = 0$, $t \geq 0$, $y(0) = 0$ whose unique solution is $y(t) = 0$. Consider obtaining an approximate solution by a simple two-step method

$$y_{j+2} + \alpha_1 y_{j+1} + \alpha_0 y_j = 0, \quad j = 0, 1, 2, \dots \quad (7.77)$$

since $\alpha_2 = 1$ and $f(t, y) = 0$. Equation (7.77) is a difference equation whose solution can be found by substituting $y_j = z^j$, $j = 0, 1, 2, \dots$ for some complex number z . The equation so obtained is

$$z^2 + \alpha_1 z + \alpha_0 = 0. \quad (7.78)$$

Since $\rho(z) = z^2 + \alpha_1 z + \alpha_0$, we see that z is a zero of $\rho(z)$. Now suppose that (7.78) has distinct roots z_1 and z_2 . (If the roots are the same, then $y_j = A_1 h z_1^j + j A_2 h z_2^j$.) Then

$$y_j = A_1 h z_1^j + A_2 h z_2^j \quad (7.79)$$

is a solution of (7.77), where A_1 and A_2 are constants.

To find A_1 and A_2 , we note that $y_0 = y_1 = 0$ which gives

$$\begin{cases} A_1 h + A_2 h = 0, \\ A_1 h z_1 + A_2 h z_2 = 0, \end{cases} \quad (7.80)$$

from which $A_1 = A_2 = 0$. Hence, (7.79) indeed gives $y_j = 0$, $j = 0, 1, 2, \dots$. However, if (7.79) is solved numerically for A_1 and A_2 or if $y_1 \neq 0$, but is very small (because y_1 may be obtained from y_0 using some single-step method in the presence of rounding errors) then nonzero A_1 and A_2 may be obtained. Now, if $|z_1|, |z_2| \leq 1$, we see that $y_j \rightarrow 0$, as $j \rightarrow \infty$, $h \rightarrow 0$, $jh = t > 0$, i.e., the method is convergent (to solution $y(t) = 0$), as it should be. However, if $|z_1| > 1$ or $|z_2| > 1$, then $|y_j| \rightarrow \infty$ as $j \rightarrow \infty$, $h \rightarrow 0$, $jh = t > 0$; the method is therefore not convergent in that case. Indeed, small errors are violently magnified as $h \rightarrow 0$. (Note, in the case $z_1 = z_2$, the solution (7.79) of (7.77) has the form $y_j = A_1 h z_1^j + j A_2 h z_1^j$ and strict inequality $|z_1| < 1$ must hold for convergence.)

We now define stability.

DEFINITION 7.11 *The k -step method (7.68) is said to be stable if the following conditions hold:*

- (a) *all roots z_j , $1 \leq j \leq k$, of $\rho(z) = 0$ satisfy $|z_j| \leq 1$,*
- (b) *if a root z_ν is not a single root, then it satisfies $|z_\nu| < 1$.*

Example 7.8

It can be shown, for example, that Euler's method, the trapezoidal method and Simpson's method are all stable. Consider Simpson's method. For Simpson's method, $\alpha_0 = -1$, $\alpha_1 = 0$, $\alpha_2 = 1$, and $\rho(z) = z^2 - 1$. Hence, $z_1 = 1$, $z_2 = -1$, and $|z_j| \leq 1$ for $j = 1, 2$. $\square$

Before continuing, a brief review may be helpful.

- A k -step method has the form

$$\sum_{l=0}^k \alpha_l y_{j+l} = h \sum_{l=0}^k \beta_l f(t_{j+l}, y_{j+l}), \quad j = 0, 1, 2, \dots, N-k \quad (7.68)$$

where $y_0, y_1, \dots, y_{k-1}$ are given. (Explicit if $\beta_k = 0$, implicit if $\beta_k \neq 0$.)

- The multistep method is consistent if the order $p \geq 1$ and thus if

$$\sum_{l=0}^k \alpha_l = 0 \quad \text{and} \quad \sum_{l=1}^k l \alpha_l = \sum_{l=1}^k \beta_l. \quad (7.74)$$

- If

$$\rho(z) = \sum_{l=0}^k \alpha_l z^l \quad \text{and} \quad \sigma(z) = \sum_{l=0}^k \beta_l z^l, \quad (7.75)$$

then the method is consistent if $\rho(1) = 0$, $\rho'(1) = \sigma(1)$.

- Furthermore, the method is stable if all roots of $\rho(z) = 0$ satisfy $|z_j| \leq 1$ with strict inequality if z_j is not a single root.

We now have the following important theorem (due to Dahlquist) connecting consistency, stability, and convergence.

THEOREM 7.5

For a multistep method of the form (7.68) to be convergent, it is necessary and sufficient for it to be consistent and stable.

PROOF We prove here only necessity for $n = 1$ although the theorem holds for $n > 1$. (In place of sufficiency, a stronger result is given in Theorem 7.6.)

If a method is convergent, then it is convergent for the initial-value problem $y'(t) = 0, t \geq 0, y(0) = 0$ whose solution is $y(t) = 0, t \geq 0$. The k -step method for this problem is:

$$y_{j+k} + \alpha_{k-1}y_{j+k-1} + \cdots + \alpha_0y_j = 0, \quad j \geq 0. \quad (7.81)$$

Since the method is convergent, for any $t > 0$ we have

$$\lim_{j \rightarrow \infty} y_j = 0, \quad (7.82)$$

where $h = t/j$, whenever

$$\lim_{h \rightarrow 0} y_j = 0, \quad 0 \leq j \leq k-1. \quad (7.83)$$

Now let $\eta = re^{i\varphi}$, $r \geq 0, 0 \leq \varphi \leq 2\pi$ be a root of $\rho(\eta) = 0$ and consider the numbers $y_0, y_1, y_2, \dots$ given by

$$y_j = hr^j \cos(j\varphi) \quad (7.84)$$

(recall that $\eta^j = r^j e^{ij\varphi} = r^j \cos(j\varphi) + ir^j \sin(j\varphi)$.) These numbers satisfy difference equation (7.81) and also satisfy (7.83). By hypothesis, (7.82) must hold for these y_j . If $\varphi = 0$ or $\varphi = \pi$, then (7.82) implies that $r \leq 1$. In case $\varphi \neq 0, \varphi \neq \pi$, note that $y_j^2 - y_{j+1}y_{j-1} = h^2 r^{2j} \cos^2(j\varphi) - h^2 r^{2j} \cos((j+1)\varphi) \cos((j-1)\varphi) = h^2 r^{2j} \sin^2(\varphi)$. That is, $\frac{y_j^2 - y_{j+1}y_{j-1}}{\sin^2(\varphi)} = h^2 r^{2j}$.

Now the left-hand side $\rightarrow 0$ as $j \rightarrow \infty$ by (7.82). This implies that $h^2 r^{2j} \rightarrow 0$ as $j \rightarrow \infty, h = t/j$ from which $r \leq 1$. We thus obtain $r \leq 1$, property (a) of the stability condition.

Now let $\eta = re^{i\varphi}$ be a root of $\rho(\eta)$ of multiplicity greater than one. Now $y_j = \sqrt{h} j r^j \cos(j\varphi)$ forms a solution of difference equation (7.81) when η is a multiple root of $\rho(\eta) = 0$ and these numbers also satisfy (7.83). If $\varphi = 0$ or $\varphi = \pi$, we obtain $|y_j| = \sqrt{h} j r^j = \sqrt{t/j} r^j \rightarrow 0$ as $j \rightarrow \infty$ for fixed

$t > 0$ only if $r < 1$. (Recall that $t = hj$.) If $\varphi \neq 0$, $\varphi \neq \pi$, we use relation $z_j^2 - z_{j+1}z_{j-1} = r^{2j} \sin^2(\varphi)$, $z_j = y_j/(j\sqrt{h})$ to obtain the same conclusion that $r < 1$. We have thus shown that convergence implies stability.

To see that convergence implies consistency, we have to show that convergence implies that $\rho(1) = 0$, $\rho'(1) = \sigma(1)$ or that $c_0 = c_1 = 0$. To show that $c_0 = 0$, consider the initial-value problem $y'(t) = 0$, $y(0) = 1$ with exact solution $y(t) = 1$. Assuming starting values $y_j = 1$, $0 \leq j \leq k-1$, we conclude that y_j , solution of (7.81), must satisfy $\lim_{j \rightarrow \infty} y_j = 1$, $jh = t > 0$ fixed. Letting

$j \rightarrow \infty$ in (7.81), we obtain that $\rho(1) = 0 = c_0 = \sum_{l=0}^k \alpha_l$. To obtain $c_1 = 0$, we consider the initial-value problem $y'(t) = 1$, $y(0) = 0$. (The exact solution is $y(t) = t$.) Thus, $y_j \rightarrow jh$ as $j \rightarrow \infty$ and $f(t_j, y_j) = 1$ for all j . Then, considering (7.68),

$$\alpha_k(j+k)h + \alpha_{k-1}(j+k-1)h + \cdots + \alpha_0(jh) = h(\beta_k + \beta_{k-1} + \cdots + \beta_0).$$

Rearranging this expression,

$$\alpha_k kh + \alpha_{k-1}(k-1)h + \cdots + jh(\alpha_k + \alpha_{k-1} + \cdots + \alpha_0) = h(\beta_k + \beta_{k-1} + \cdots + \beta_0).$$

Hence, $\rho'(1) = \sigma(1)$, because $(\alpha_k + \alpha_{k-1} + \cdots + \alpha_0) = 0$. $\square$

Consider what happens when a method fails to be consistent or fails to be stable.

First consider the method:

$$y_{j+2} + 4y_{j+1} - 5y_j = h(4f_{j+1} + 2f_j) \quad (7.85)$$

This method is consistent with order $p = 3$ but it is not stable, i.e., the roots of $z^2 + 4z - 5 = 0$ are $z = 1$ and $z = -5$. By Theorem 7.5, it is not convergent. Apply the method to $y' = 4t\sqrt{y}$, $0 \leq t \leq 2$, $y(0) = 1$ with solution $y(t) = (1 + t^2)^2$. Using exact starting values in (7.85), $y_0 = 1$, $y_1 = (1 + h^2)^2$, we obtain with $h = 0.1$,

t	0	0.2	0.4	1.0	1.1	2.0
exact solution	1	1.082	1.346	4.000	4.884	25.000
numerical solution	1	1.081	1.339	-68.69	367.26	-6.96×10^8

The errors are oscillating wildly.

Now consider the following method:

$$y_{j+2} - y_{j+1} = \frac{h}{3}(3f_{j+1} - 2f_j) \quad (7.86)$$

This two-step method is stable but it is not consistent. We have $\rho(z) = 0 - z + z^2$, $\sigma(z) = -\frac{2}{3} + \frac{2}{3}z + 0z^2$, $\rho(1) = 0$, but $\rho'(z) = -1 + 2z$, $\rho'(1) = 1 \neq \sigma(1) = \frac{2}{3}$. For the same initial-value problem as above, we obtain:

t	0	0.1	0.2	1.0	2.0
exact solution	1	1.020	1.082	4.000	25.000
$h = 0.1$	1	1.020	1.060	2.075	6.803
$h = 0.05$	1	1.015	1.045	1.932	6.141

As the step size decreases, the error increases rather than decreases, although the violent behavior, characteristic of unstable methods, is absent.

It is of course desirable to use a stable method with high order p . (With high-order accuracy, the number of time steps can be reduced to decrease rounding errors.) We have the following error estimate for multistep k -step methods.

THEOREM 7.6

Suppose that the k -step method (7.68) is stable and of order $p \geq 1$ (and thus consistent). Let the conditions of Theorem 7.2 be satisfied and let $y(t)$, the solution of IVP (7.1) (for $n = 1$) be $p + 1$ times continuously differentiable on $[a, b]$. Then there is an $h_0 > 0$ such that $0 < h \leq h_0$ and the estimate

$$\max_{0 \leq j \leq N} |y_j - y(t_j)| \leq c \left\{ \max_{0 \leq j \leq k-1} |y_j - y(t_j)| + h^p \max_{a \leq t \leq b} |y^{(p+1)}(t)| \right\} \quad (7.87)$$

holds for some positive constant c independent of $y(t)$, y_j , and h .

PROOF See [90]. □

We now turn to absolute stability. A numerical method may be stable and consistent (hence convergent) but when applied to certain problems may require step size h too small for practical consideration. (See the example preceding Definition 7.5.) Small values of h will also lead to larger rounding errors. Our definition of absolute stability applied to multistep method is:

DEFINITION 7.12 The multistep method (7.68) is said to be absolutely stable if applied to the scalar equation $y' = \lambda y$, $t \geq 0$, yields values $\{y_j\}_{j \geq 0}$ which satisfy $y_j \rightarrow 0$ as $j \rightarrow \infty$. The set of values λh for which $y_j \rightarrow 0$ as $j \rightarrow \infty$ is called the set of absolute stability of (7.68).

We have the following result:

PROPOSITION 7.1

The k -step multistep method (7.68) is absolutely stable provided that the roots z_j , $1 \leq j \leq k$, of the polynomial $p(z, \lambda h) = \rho(z) - \lambda h \sigma(z)$ satisfy $|z_j| < 1$.

PROOF Applying the k -step method (7.68) to $y' = \lambda y$, we obtain

$$\sum_{l=0}^k (\alpha_l - h\lambda\beta_l)y_{j+l} = 0, \quad 0 \leq j \leq k-1.$$

The solution of this difference equation has the form:

$$y_j = \sum_{m=1}^k \mu_m z_m^j,$$

where for each m , $1 \leq m \leq k$.

$$\sum_{l=0}^k (\alpha_l - \lambda h\beta_l)z_m^l = 0.$$

Thus, we require the roots of $p(z, \lambda h) = \rho(z) - \lambda h\sigma(z) = \sum_{l=0}^k (\alpha_l - \lambda h\beta_l)z^l$ to satisfy $|z_j| < 1$, $1 \leq j \leq k$. $\square$

Example 7.9

(Extremes)

(a) *Midpoint Multistep Method* has the form

$$y_{j+2} - y_j = 2hf_{j+1}$$

and thus $\beta_0 = 0$, $\beta_1 = 2$, $\beta_2 = 0$, $\sigma_0 = -1$, $\sigma_1 = 0$, $\sigma_2 = 1$. Since $k = 2$, $\rho(z) = -1 + z^2$ and $\sigma(z) = 2z$. (Notice first that the method is stable and consistent and $\rho(1) = 0$, $\rho'(1) = \sigma(1)$, and the roots of $\rho(z) = 0$ are simple with magnitude ≤ 1 .) We have $p(z, \lambda h) = -1 + z^2 - 2\lambda hz$, which has roots $z = \lambda h \pm \sqrt{(\lambda h)^2 + 1}$, so $z_1 = \lambda h + \sqrt{(\lambda h)^2 + 1} > 1$ for any $\lambda h > 0$. Consider $\lambda h < 0$. Then $z_2 = \lambda h - \sqrt{(\lambda h)^2 + 1} < -1$ for any $\lambda h < 0$. Thus, the midpoint method is not absolutely stable for any real λh .

(b) *The Trapezoidal Method* (7.69) has the form

$$y_{j+1} - y_j = h\left(\frac{1}{2}f_{j+1} + \frac{1}{2}f_j\right)$$

and hence $\rho(z) = -1 + z$ and $\sigma(z) = \frac{1}{2}(1 + z)$, and $p(z, \lambda h) = -1 + z - \lambda h(\frac{1}{2}(1 + z))$ has root $z_1 = \frac{1 + \frac{\lambda h}{2}}{1 - \frac{\lambda h}{2}}$. Thus, for all $\operatorname{Re}(\lambda h) < 0$, we see that $|z_1| < 1$. The set of absolute stability for the trapezoidal method, if λ is complex, is the open left-half complex plane, and if λ is real, the interval of absolute stability is $(-\infty, 0)$. (Note that the trapezoidal method is consistent and stable as $\rho(1) = 0$, $\rho'(1) = \sigma(1)$, and the roots of $\rho(z)$ are simple with magnitude ≤ 1 .)

□

REMARK 7.11 Multistep methods have the advantage over Runge–Kutta methods of providing high accuracy with few function evaluations. For example, the four-step explicit Adams–Bashforth method has the form

$$y_{j+4} = y_{j+3} + \frac{h}{24} [55f(t_{j+3}, y_{j+3}) - 59f(t_{j+2}, y_{j+2}) + 37f(t_{j+1}, y_{j+1}) - 9f(t_j, y_j)] \quad (7.88)$$

where $y_0 = y(t_0)$ and y_1, y_2, y_3 are obtained using a 4-*th*-order Runge–Kutta method. This method only requires one new function evaluation per time step, namely, $f(t_{j+3}, y_{j+3})$. (Recall that the 4-*th*-order Runge–Kutta method requires 4 new function evaluations per time step.) □

7.6 Predictor–Corrector Methods

We now consider predictor–corrector methods. Consider an implicit method of the form:

$$y_{j+k} + \sum_{l=0}^{k-1} \alpha_l y_{j+l} = h\beta_k f(t_{j+k}, y_{j+k}) + h \sum_{l=0}^{k-1} \beta_l f_{j+l} \quad (7.89)$$

where ($\beta_k \neq 0$). A simple iterative scheme for solution of (7.89) is to compute the sequence $y_{j+k}^{(s)}$, $s \geq 0$ defined by

$$\begin{cases} y_{j+k}^{(0)} = y_{j+k-1} \\ y_{j+k}^{(s+1)} + \sum_{l=0}^{k-1} \alpha_l y_{j+l} = h\beta_k f(t_{j+k}, y_{j+k}^{(s)}) + h \sum_{l=0}^{k-1} \beta_l f_{j+l} \end{cases} \quad (7.90)$$

for $s = 0, 1, 2, \dots$. Provided that $Lh|\beta_k| < 1$, this fixed-point iteration sequence $\{y_{j+k}^{(s)}\}_{s \geq 0}$ converges to y_{j+k} , the exact solution of nonlinear system (7.89). If $y_{j+k}^{(0)}$ is near y_{j+k} , then the iterations may converge rapidly.

A way to predict $y_{j+k}^{(0)}$ accurately is to use an explicit multistep method as a *predictor*, then use the implicit method (7.89) as a *corrector*. A predictor–corrector method has the form:

$$\text{Predictor } P: y_{j+k}^{(0)} + \sum_{l=0}^{k-1} \alpha_l^* y_{j+l} = h \sum_{l=0}^{k-1} \beta_l^* f_{j+l} \quad (7.91)$$

$$\text{Corrector } C: y_{j+k}^{(s+1)} + \sum_{l=0}^{k-1} \alpha_l y_{j+l} = h\beta_k f(t_{j+k}, y_{j+k}^{(s)}) + h \sum_{l=0}^{k-1} \beta_l f_{j+l} \quad (7.92)$$

for $s = 0, 1, 2, \dots$.

The question arises as when do we stop the iterations. One could correct until $\|y_{j+k}^{(s+1)} - y_{j+k}^{(s)}\| < \epsilon$. However, usually (7.91) is applied once and (7.92) is applied m times (generally $m = 1$ or $m = 2$).

Some examples of predictor-corrector pairs are: *Euler-Trapezoidal Method*:

$$\begin{cases} \text{(P)} & y_{j+1} - y_j = hf_j & (p = 1) \\ \text{(C)} & y_{j+1} - y_j = \frac{h}{2}(f_{j+1} + f_j) & (p = 2) \end{cases} \quad (7.93)$$

Adam's Method:

$$\begin{cases} \text{(P)} & y_{j+4} - y_{j+3} = \frac{h}{24}(55f_{j+3} - 59f_{j+2} + 37f_{j+1} - 9f_j) & (p = 4) \\ \text{(C)} & y_{j+4} - y_{j+3} = \frac{h}{24}(9f_{j+4} + 19f_{j+3} - 5f_{j+2} + f_{j+1}) & (p = 4) \end{cases} \quad (7.94)$$

Adam's method uses a 4-step Adams-Bashforth method as the predictor and a 3-step Adams-Moulton method as the corrector.

Example of Euler-Trapezoidal Method

Consider

$$\begin{cases} y'(t) = t + y \\ y(1) = 2 \end{cases}$$

with exact solution $y(t) = -t - 1 + 4e^{-1}e^t$. Let $h = 0.1$ with $y_0 = 2$ and $t_0 = 1$. Let $m = 1$ (one correction). Then, for this problem,

$$\begin{cases} y_{j+1}^{(0)} = y_j + h(t_j + y_j) \\ y_{j+1} = y_j + \frac{h}{2}(t_{j+1} + y_{j+1}^{(0)}) + \frac{h}{2}(t_j + y_j) \end{cases}$$

The following numerical results are obtained:

j	t_j	y_j	$y(t_j)$
0	1	2	2
1	1.1	2.32	2.32068

Some properties of Predictor-Corrector pairs are now summarized.

(i) *Order of accuracy*

Let p^* be the order of accuracy of the predictor and p the order of accuracy of the corrector. If we correct $m \geq 1$ times and if $p^* \geq p - 1$, then the order of accuracy of the PC pair (7.91)–(7.92) is p , the order of the corrector. For example, the Euler–Trapezoidal pair with $p^* = 1$ and $p = 2$ will have order $p = 2$. For Adam's method, we have $p^* = p = 4$ and hence the predictor-corrector pair has order $p = 4$.

(ii) *Stability*

The stability properties of the PC pair (7.91)–(7.92) are the same as those of the corrector. (Thus, we obtain the generally superior stability properties of the implicit corrector method.)

(iii) *Absolute stability*

Absolute stability of the PC pair generally depends on m and PC pair. For the Adam's pair, the interval of absolute stability is $(-3, 0)$ if we "correct to convergence," i.e., as $m \rightarrow \infty$. If $m = 1$, the interval of absolute stability for the Adams pair is $(-1.25, 0)$. (Interval of absolute stability for the Euler–Trapezoidal pair is $(-2, 0)$.)

REMARK 7.12 As in the Runge–Kutta method, it is possible to devise strategies to control local error by varying step size. Good programs are available based on predictor-corrector methods with variable step size and estimation of local error. $\square$

REMARK 7.13 Tabulated below are intervals of absolute stability of several methods.

	Interval
Fourth-Order Runge–Kutta	$(-2.78, 0)$
Adams–Bashforth, Adams–Moulton PC($m = 1$)	$(-1.25, 0)$
Adams–Bashforth, Adams–Moulton PC($m = \infty$)	$(-3.00, 0)$
Gragg's Method (extrapolation)	$(-3.10, 0)$
Trapezoidal Rule (solved exactly at each step)	$(-\infty, 0)$

 $\square$

7.7 Stiff Systems

Consider the initial-value problem $y : [0, \infty) \rightarrow \mathbb{R}^n$ such that

$$y'(t) = Ay(t), \quad t \geq 0, \quad y(0) = y_0. \quad (7.95)$$

For now,¹¹ we will assume that A is an $n \times n$ matrix with *simple* eigenvalues¹² λ_i , $1 \leq i \leq n$, with corresponding eigenvectors x_i , $1 \leq i \leq n$. It is well known

¹¹We will consider the simplest case here.

¹²that is, with n distinct eigenvalues and, hence, with n linearly independent eigenvectors

that the explicit solution of (7.95) is

$$y(t) = \sum_{i=1}^n c_i e^{\lambda_i t} x_i, \quad (7.96)$$

where c_i , $1 \leq i \leq n$ are constants found by letting $t = 0$ in (7.96) and solving

$$y_0 = \sum_{i=1}^n c_i x_i$$

for the c_i 's. (You will derive this in Exercise 31 on page 437.)

The term *stiff system* originated in the study of mechanical systems with springs. A spring is “stiff” if its spring constant is large; in such a mechanical system, the spring will cause motions of the system that are fast relative to the time scale on which we are studying the system. In the numerical solution of initial value problems, “stiffness” has come to mean that the solution to the ODE has some components that vary or die out rapidly in relation to the other components, or in relation to the time interval over which the integration proceeds. For example, the scalar equation $y' = -1000y$ might be considered to be moderately stiff when it is integrated for $0 \leq t \leq 1$.

Example 7.10

Let's consider a stiff system

$$y' = Ay, \quad t \geq 0, \quad y(0) = (1, 0, -1)^T \quad (7.97)$$

where

$$A = \begin{pmatrix} -21 & 19 & -20 \\ 19 & -21 & 20 \\ 40 & -40 & -40 \end{pmatrix}.$$

The eigenvalues of A are $\lambda_1 = -2$, $\lambda_2 = -40 + 40i$ and $\lambda_3 = -40 - 40i$ and the exact solution of (7.97) is

$$\begin{cases} y_1(t) = \frac{1}{2}e^{-2t} + \frac{1}{2}e^{-40t}(\cos 40t + \sin 40t), \\ y_2(t) = \frac{1}{2}e^{-2t} - \frac{1}{2}e^{-40t}(\cos 40t + \sin 40t), \\ y_3(t) = -e^{-40t}(\cos 40t - \sin 40t), \end{cases} \quad (7.98)$$

with graphs as in Figure 7.4. Notice that for $0 \leq t \leq .1$, $y_i(t)$, $1 \leq i \leq 3$, vary rapidly but for $t \geq 0.1$, then y_i vary slowly. Hence, a small time step must be used in the interval $[0, 0.1]$ for adequate resolution whereas for $t \geq 0.1$ large time steps *should* suffice. Suppose we use Euler's method starting at $t = 0.2$

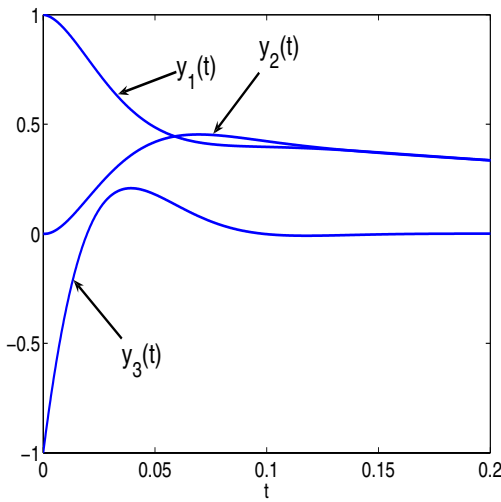


FIGURE 7.4: Exact solution for the stiff ODE system given in Example 7.10.

with initial conditions taken as the exact values $y_i(0.2)$, $1 \leq i \leq 3$. We obtain:

For $h = 0.04$

j	t_j	y_{1j}	y_{2j}	y_{3j}
0	0.2	0.335	0.335	-0.00028
5	0.4	0.218	0.223	0.0031
10	0.6	0.186	.106	-0.0283
15	0.8	-0.519	0.711	0.1436
20	1.0	9.032	-8.91	1.9236
21	1.1	-6.862	6.98	27.55

For $h = 0.02$

j	t_j	y_{1j}	y_{2j}	y_{3j}
0	0.2	0.3353	0.3350	-0.00028
5	0.3	0.2734	0.2732	-0.000065
10	0.4	0.2229	0.2228	-0.0000054

Violent instability occurs for $h = 0.04$ but the method is stable for $h = 0.02$. What happened? Why do we need h so small? $\square$

The answers lie in understanding absolute stability.

7.7.1 Absolute Stability of Methods for Systems

Earlier (Definition 7.5 on page 400), we defined absolute stability of methods for solving the IVP in terms of the scalar equation $y' = \lambda y$. We now extend the definition to systems.

DEFINITION 7.13 *Let A satisfy the stated assumptions and suppose that $\operatorname{Re} \lambda_i < 0$, $1 \leq i \leq n$. A numerical method for solving IVP (7.95) is called absolutely stable for a particular value of the product λh if it yields numerical solutions, y_j , $j \geq 0$, in $\mathbb{R}^n$ such that $y_j \rightarrow 0$ as $j \rightarrow \infty$ for all y_0 . As in Definition 7.5, we speak of the region of absolute stability as being the set of λh in the complex plane for which the method, applied to a scalar equation $y' = \lambda y$, $y \in \mathbb{R}$, is absolutely stable. (The reason this will make sense for systems will become apparent below.)*

Notice that a method for a system is absolutely stable if and only if the method is absolutely stable for the scalar equations $z' = \lambda_i z$, for $1 \leq i \leq n$. To see this, consider, for example, the k -step method

$$\sum_{l=0}^k \alpha_l y_{l+j} = h \sum_{l=0}^k \beta_l f_{l+j} = h \sum_{l=0}^k \beta_l A y_{l+j}.$$

Thus,

$$\sum_{l=0}^k (\alpha_l I - h \beta_l A) y_{l+j} = 0, \quad j \leq 0.$$

Let $P^{-1}AP = \Lambda$, where this decomposition is guaranteed if A has n simple eigenvalues and $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$. We conclude that

$$\sum_{l=0}^k (\alpha_l I - h \beta_l \Lambda) P^{-1} y_{l+j} = 0.$$

Setting $z_j = P^{-1}y_j$, we see that

$$\sum_{l=0}^k (\alpha_l - h \beta_l \lambda_i) (z_{l+j})_i = 0, \quad 1 \leq i \leq n,$$

where $(z_{l+j})_i$ is the i -th component of z_{l+j} . Since $(z_j)_i \rightarrow 0$, $1 \leq i \leq n$, as $j \rightarrow \infty$ if and only if $y_j \rightarrow 0$ as $j \rightarrow \infty$, we see that the method will be absolutely stable for system (7.95) if and only if it is absolutely stable for the scalar equation $z' = \lambda_i z$, $1 \leq i \leq n$. In this case, it will be absolutely stable provided that the roots of $p(z, h; i) = \rho(z) - h \lambda_i \sigma(z)$, $1 \leq i \leq n$, satisfy $|z_{l,i}| < 1$, $1 \leq l \leq k$, $1 \leq i \leq n$.

For Euler's method, we found that the region of absolute stability is the open disk

$$\{\lambda h : |1 + \lambda h| < 1\} \quad (7.99)$$

(See Figure 7.5.) (Recall $y_{j+1} = y_j + \lambda h y_j$ for Euler's method applied to

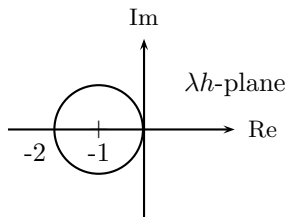


FIGURE 7.5: The open disc representing the stability region for Euler's method.

$y' = \lambda y$ gives $y_j \rightarrow 0$ if $|1 + \lambda h| < 1$.)

Applying this reasoning to Example 7.10, we see that, for the numerical solutions to go to zero (that is, for absolute stability), we must have $|1 + \lambda_i h| < 1$, $1 \leq i \leq 3$. For $i = 1$ ($\lambda_1 = -2$), this yields $h < 1$. However, $i = 2, 3$ ($\lambda_2 = -40 + 40i$, $\lambda_3 = -40 - 40i$) yields $h < 1/40 = .025$ which is violated if $h = .04$. We conclude that, although the terms with eigenvalues λ_2 , λ_3 contribute almost nothing to the solution of (7.97) after $t = .1$, they *force* the selection of small time step h which must satisfy $|1 + \lambda_2 h| < 1$, $|1 + \lambda_3 h| < 1$.

7.7.2 Methods for Stiff Systems

Recapitulating, we have

DEFINITION 7.14 *The linear system (7.95) is said to be stiff if $\operatorname{Re} \lambda_i < 0$, $1 \leq i \leq n$, and $|\operatorname{Re} \lambda_\mu| = \max_{1 \leq i \leq n} |\operatorname{Re} \lambda_i| \gg |\operatorname{Re} \lambda_\nu| = \min_{1 \leq i \leq n} |\operatorname{Re} \lambda_i|$, or if $\max_{1 \leq i \leq n} |\operatorname{Re} \lambda_i|$ is large in relation to the length $b - a$ of the interval of integration.*

REMARK 7.14 Note that the stiff system (7.95) is a model for nonlinear systems $y' = f(t, y)$ with large Lipschitz constant L . The matrix A is a model for the Jacobi matrix $\partial f / \partial y$, i.e., expanding in a Taylor series about fixed $\tilde{y}$,

$$f(t, y) \approx f(t, \tilde{y}) + \frac{\partial f}{\partial y}(t, \tilde{y})(y - \tilde{y}).$$

□

We now study selection of methods suitable for nonlinear stiff systems. First,

DEFINITION 7.15 *A numerical method is called A-stable if its region of stability contains all of the open left-half plane $\operatorname{Re}(\lambda h) < 0$.*

REMARK 7.15 Note that, if a method is A-stable, then the numerical solution based on that method will go to zero (and thus behave qualitatively like the true solution) regardless of how we choose h . $\square$

The trapezoidal method

$$y_{j+1} = y_j + \frac{h}{2}(f_j + f_{j+1})$$

has region of absolute stability $\operatorname{Re}(\lambda h) < 0$, and therefore the trapezoidal method is A-stable. Unfortunately, the trapezoidal method is implicit and has order of accuracy $p = 2$. However, we have the following theorem of Dahlquist.

THEOREM 7.7

*The order of any A-stable implicit multistep method is always $p \leq 2$. Moreover, there exists no A-stable explicit multistep method.*¹³

Thus, the trapezoidal method is popular for solving stiff problems, even though the order is $p = 2$, since no multistep methods are A-stable with $p > 2$. The trapezoidal method has the form

$$y_{j+1} = y_j + \frac{h}{2}f(t_{j+1}, y_{j+1}) + \frac{h}{2}f(t_j, y_j),$$

which is generally solved for y_{j+1} by applying a few iterations (normally three or four) of Newton's method to the nonlinear system to obtain y_{j+1} , i.e., we need to determine y such that $F(y) = 0$, where

$$F(y) = y - y_j - \frac{h}{2}f(t_j, y_j) - \frac{h}{2}f(t_{j+1}, y).$$

We will see that Newton's method is

$$w_{l+1} = w_l - (F'(w_l))^{-1}F(w_l)$$

where $F'(w_l)$ is the Jacobian matrix.

¹³This is in G. Dahlquist, "A Special Stability Problem for Linear Multistep Methods," *BIT* **3** (1963), pp. 27–43.

There are other popular methods for solving nonlinear stiff systems. Non-A-stable methods, but with large regions of stability, have been developed by C. W. Gear and others which use implicit multistep methods or implicit Runge–Kutta methods [51]. An example of an implicit Runge–Kutta method originally proposed by Hammer and Hollingworth is

$$\begin{cases} y_{j+1} = y_j + \gamma_1 K_1 + \gamma_2 K_2 \\ K_1 = hf(t_j + \alpha_1 h, y_j + \beta_{11} K_1 + \beta_{12} K_2) \\ K_2 = hf(t_j + \alpha_2 h, y_j + \beta_{21} K_1 + \beta_{22} K_2), \end{cases} \quad (7.100)$$

where

$$\begin{aligned} \gamma_1 &= \gamma_2 = \frac{1}{2}, & \alpha_1 &= \frac{1}{2} + \frac{\sqrt{3}}{6}, & \alpha_2 &= \frac{1}{2} - \frac{\sqrt{3}}{6}, \\ \beta_{11} &= \beta_{22} = \frac{1}{4}, & \beta_{12} &= \frac{1}{4} - \frac{\sqrt{3}}{6}, & \beta_{21} &= \frac{1}{4} + \frac{\sqrt{3}}{6}. \end{aligned}$$

This method has order $p = 4$ and requires solution of $2n$ nonlinear equations in $2n$ unknowns per time step. The interval of absolute stability of method (7.100) can be shown to be $(-\infty, 0)$, the same as the trapezoidal rule, but with order twice that of the trapezoidal rule.

It is worthwhile to consider one other type of method that is suitable for stiff systems, *Padé methods*. Recall that we are considering, for determination of absolute stability, numerical approximation of

$$y' = \lambda y. \quad (7.101)$$

Notice now that the trapezoidal method for $y' = \lambda y$ can be written as

$$y_{j+1} = \left(\frac{1 + \frac{\lambda h}{2}}{1 - \frac{\lambda h}{2}} \right) y_j,$$

and it is easily verified that $(1 + \frac{\lambda h}{2})/(1 - \frac{\lambda h}{2})$ is an $\mathcal{O}((\lambda h)^3)$ approximation to $e^{\lambda h}$. Typically, a *single-step* method applied to $y' = \lambda y$, $y(0) = y_0$ (whose exact solution is $y = e^{\lambda t} y_0$) gives approximations which satisfy $y_{j+1} = R(\lambda h) y_j$ or $y_{j+1} = (R(\lambda h))^j y_0$, where $(R(\lambda h))^j$ approximates $e^{\lambda h j}$, and hence $R(\lambda h)$ approximates $e^{\lambda h}$. Thus, $R(z)$ must be an approximation to e^z for $|z|$ sufficiently small.

Now recall Padé approximations to e^z . Let $R_{m,n}(z)$ be the (m, n) Padé approximation to e^z of the form $p(z)/q(z)$, where $p \in \mathcal{P}_m$, $q \in \mathcal{P}_n$ and $R_{m,n}^{(k)}(0) = 1$ for $k = 0, 1, 2, \dots, m+n$. That is, the derivatives of the rational approximation $R_{m,n}(z)$ agree with those of e^z at $z = 0$ up to order $m+n$. We saw earlier that $R_{m,n}(z)$ is good approximation to e^z and $R_{m,n}(z) = e^z + \mathcal{O}(z^{m+n+1})$.

Some of the first few Padé approximations to e^z are given in Table 7.1. Note that $R_{1,0}$ corresponds to Euler's method, $R_{0,1}$ corresponds to the backward

TABLE 7.1: Padé approximations to e^z

n	m		
	0	1	2
0	1	$1 + z$	$1 + z + \frac{z^2}{2}$
1	$\frac{1}{1 - z}$	$\frac{1 + z/2}{1 - z/2}$	$\frac{6 + 4z + z^2}{6 - 2z}$
2	$\frac{1}{1 - z + z^2/2}$	$\frac{6 + 2z}{6 - 4z + z^2}$	$\frac{12 + 6z + z^2}{12 - 6z + z^2}$

Euler method, and $R_{1,1}$ corresponds to the trapezoidal method. We have the following notes on Padé methods. To this end, we first present another definition of stability.

DEFINITION 7.16 A method is $A(0)$ -stable if there is a $\theta \in (0, \frac{\pi}{2})$ such that the region of absolute stability contains the infinite wedge

$$S_\theta = \{\lambda h \mid -\theta < \pi - \arg(\lambda h) < \theta\}.$$

(See Figure 7.6.)

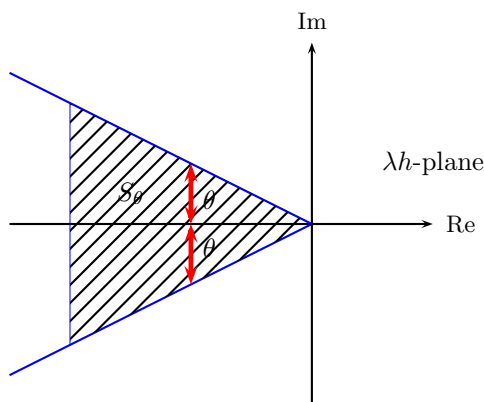


FIGURE 7.6: Illustration of the region of absolute stability for $A(0)$ -stability (Definition 7.16).

(Notice that if a method is A -stable then it is $A(0)$ -stable.)

REMARK 7.16 The following classical result holds for Padé methods:

The methods corresponding to $n \geq m$ in the Padé table are A(0)-stable and the methods corresponding to $n = m$, $n = m + 1$, or $n = m + 2$ are A-stable. $\square$

REMARK 7.17 Only the Padé methods with $m \leq 1$, $n \leq 1$ are linear multistep methods. All the Padé methods are single-step methods and are explicit if $n = 0$ and are implicit if $n \geq 1$. The order of the Padé method, corresponding to values m and n in the Padé table is $\mathcal{O}(h^{m+n})$. $\square$

Example 7.11

Consider the Padé method corresponding, for example, to $m = 1$ and $n = 2$. We induce the corresponding Padé method by replacing $f(t, y) = \lambda y$ by the general $f(t, y)$. Beginning with the Padé approximation to $e^{\lambda t}$, we have

$$e^{\lambda(t+h)} = e^{\lambda t} e^{\lambda h} \approx e^{\lambda t} \left\{ \frac{6 + 2(\lambda h)}{6 - 4(\lambda h) + (\lambda h)^2} \right\}, \quad \text{whence}$$

$$e^{\lambda t+h} \{6 - 4(\lambda h) + (\lambda h)^2\} \approx e^{\lambda t} \{6 + 2(\lambda h)\}.$$

Replacing $e^{\lambda t}$ by y_j , $e^{\lambda(t+h)}$ by y_{j+1} , $e^{\lambda t} \lambda$ by $y'(t_j) = f(t_j, y_j)$, $e^{\lambda(t+h)} \lambda$ by $y'(t_{j+1}) = f(t_{j+1}, y_{j+1})$, and $e^{\lambda(t+h)} \lambda^2$ by

$$y''(t_{j+1}) = f'(t_{j+1}, y_{j+1}) = \frac{\partial f}{\partial t}(t_{j+1}, y_{j+1}) + \frac{\partial f}{\partial y}(t_{j+1}, y_{j+1}) f(t_{j+1}, y_{j+1}),$$

we obtain

$$6y_{j+1} - 4hf(t_{j+1}, y_{j+1}) + h^2 f'(t_{j+1}, y_{j+1}) = 6y_j + 2hf(t_j, y_j).$$

This simplifies to

$$y_{j+1} = y_j + \frac{h}{3} f(t_j, y_j) + \frac{2h}{3} f(t_{j+1}, y_{j+1}) - \frac{h^2}{6} f'(t_{j+1}, y_{j+1}).$$

It is straightforward to show that the local truncation error for this one-step method is $\mathcal{O}(h^4)$ and hence the global error is $\mathcal{O}(h^3)$. In addition, this method is A-stable. (Exercise 32 on page 437) $\square$

7.8 Extrapolation Methods

Recall the Richardson extrapolation process. Suppose that $A_1(h)$ is an approximation to a number A which depends on a parameter h such as step size. Suppose that the error in the approximation satisfies

$$A - A_1(h) = c_1 h + c_2 h^2 + \cdots + c_N h^N + \mathcal{O}(h^{N+1}) \quad (7.102)$$

for some $N \geq 1$ where $c_1, c_2, \dots, c_N$ are constants independent of h . Furthermore,

$$A - A_1\left(\frac{h}{2}\right) = c_1 \frac{h}{2} + c_2 \left(\frac{h}{2}\right)^2 + \cdots + c_N \left(\frac{h}{2}\right)^N + \mathcal{O}(h^{N+1}) \quad (7.103)$$

Multiplying (7.103) by 2 and subtracting (7.102) we obtain

$$A - \left(2A_1\left(\frac{h}{2}\right) - A_1(h)\right) = c_2' h^2 + c_3' h^3 + \cdots + c_N' h^N + \mathcal{O}(h^{N+1}) \quad (7.104)$$

Thus, $2A_1(\frac{h}{2}) - A_1(h)$ should approximate A better than $A_1(h)$ or $A_1(\frac{h}{2})$ for small h .

Assuming an error expansion of the form (7.102), the process can be continued N times. For notational convenience, let $A_2(h) = 2A_1(\frac{h}{2}) - A_1(h)$. We can then put the extrapolation procedure in the tabular form:

	$A_1(h)$				
	$A_1(\frac{h}{2})$	$A_2(h)$			
	$A_1(\frac{h}{4})$	$A_2(\frac{h}{2})$	$A_3(h)$		
	$A_1(\frac{h}{8})$	$A_2(\frac{h}{4})$	$A_3(\frac{h}{2})$	$A_4(h)$	
	$A_1(\frac{h}{16})$	$A_2(\frac{h}{8})$	$A_3(\frac{h}{4})$	$A_4(\frac{h}{2})$	$A_5(h)$
	$\vdots$				
Accuracy	$\mathcal{O}(h)$	$\mathcal{O}(h^2)$	$\mathcal{O}(h^3)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^5)$

where $A_j(h) = \frac{2^{j-1}A_{j-1}(\frac{h}{2}) - A_{j-1}(h)}{2^{j-1} - 1}$.

We now consider two numerical methods for solution of IVP (7.1) that have error expansion of the correct form for applying Richardson extrapolation. One such method is Euler's method. (It is interesting that Euler's method is a Runge-Kutta method, a Taylor series method, a one-step multistep method, and also a method adaptable to extrapolation.)

Euler's method has the form:

$$\begin{cases} y_{j+1} = y_j + hf(t_j, y_j), & 0 \leq j \leq N-1 \\ y_0 = y_a. \end{cases} \quad (7.105)$$

For any time t_{j+1} , the error has the form

$$y_{j+1} - y(t_{j+1}) = hc_1(t_{j+1}) + h^2c_2(t_{j+1}) + h^3c_3(t_{j+1}) + \cdots \quad (7.106)$$

and thus Richardson extrapolation can be employed.

REMARK 7.18 It is generally quite difficult to prove that the error has an expansion of the correct form. For a comprehensive treatment of Richardson extrapolation and error expansions, see [56]. □

REMARK 7.19 The extrapolation procedure can be applied at each time step to reduce local error to below a given tolerance in an error control method. Suppose that y_j is a good approximation to $y(t_j)$, to find y_{j+1} , the following table is computed:

step	Euler	Extrapolations		
h	$w_{1,1}$			
$\frac{h}{2}$	$w_{2,1}$	$w_{2,2}$		
$\frac{h}{4}$	$w_{3,1}$	$w_{3,2}$	$w_{3,3}$	
$\frac{h}{8}$	$w_{4,1}$	$w_{4,2}$	$w_{4,3}$	$w_{4,4}$

where $w_{k,1}$ is the Euler approximation to $y(t_{j+1})$ using step $\frac{h}{2^{k-1}}$ with the initial value y_j . When $\|w_{l,l} - w_{l-1,l-1}\| < \epsilon$, then $y_{j+1} = w_{l,l}$. □

Example 7.12
($n = 1$)

$$\begin{cases} y'(t) = t + y \\ y(1) = 2 \end{cases}$$

(exact solution $y(t) = -t - 1 + 4e^{-1}e^t$). Let $\epsilon = 10^{-4}$ and initial $h = 0.1$ and $y_0 = 2$.

h	Euler	Extrapolations		
0.1	2.3			
0.05	2.31	2.3200		
0.025	2.315252	2.3205	2.32067	
0.0125	2.317944	2.32064	2.32068	2.32068 (stop)

Set $y_1 = 2.32068$. Now compute y_2 similarly. □

Generally, recalling Romberg integration, Richardson extrapolation is much more effective when the error expansion is of even order, i.e.,

$$A - A_1(h) = c_2h^2 + c_4h^4 + c_6h^6 + c_{2N}h^{2N} + \mathcal{O}(h^{2N+1}) \tag{7.107}$$

where now

$$A_j(h) = \frac{4^{j-1}A_{j-1}(\frac{h}{2}) - A_{j-1}(h)}{4^{j-1} - 1}.$$

However, it is very difficult to find *explicit* methods with even-order error expansions. Gragg's method is one such method.

In Gragg's method, Euler's method is applied first, then the 2-step midpoint method is applied, and finally a smoothing step is applied. Gragg's method has the form:

$$\left\{ \begin{array}{ll} w_1 &= w_0 + hf(t_0, w_0) & \text{Euler first step} \\ w_{i+1} &= w_{i-1} + 2hf(t_i, w_i), \quad i = 1, 2, \dots, M & \text{midpoint method} \\ w_{M+1} &= \frac{1}{4}w_{M+1} + \frac{1}{2}w_M + \frac{1}{4}w_{M-1} & \text{smoothing step.} \end{array} \right. \quad (7.108)$$

Then $w_{M+1} \approx y(t_M)$, $t_M = Mh + t_0$

Example 7.13

$$\begin{cases} y'(t) = t + y, \\ y(1) = 2. \end{cases}$$

(The exact solution is $y(t) = -t - 1 + 4e^{-1}e^t$ and $y(1.1) = 2.3206837$.) Let $\epsilon = 10^{-4}$ and initial $h = 0.1$.

h	$w_{M+1} \approx y(1.1)$		
0.1	2.32		
0.05	2.3205	2.32067	
0.025	2.32063766	2.32068	2.3206847
Accuracy	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$

Now set $y_1 = 2.32068$ and compute y_2 similarly. □

REMARK 7.20 A variation of Gragg's method called the GBS method (obtained by applying rational extrapolation to Gragg's method) is one of the most efficient general purpose methods for numerical solution of initial-value problems [50]. In a survey in 1971, the GBS method emerged as the best method of those tested when function evaluations were relatively inexpensive (when each function evaluation required less than 25 arithmetic operations). □

REMARK 7.21 For more information on numerical methods for initial-value problems see, for example, [37], [50], [83], [84], or [90]. □

7.9 Application to Parameter Estimation in Differential Equations

Many phenomena in the biological and physical sciences have been described by parameter-dependent systems of differential equations such as those discussed previously in this chapter. Furthermore, some of the parameters in these models cannot be directly measured from observed data. Thus, parameter estimation techniques discussed in this section are crucial in order to use such differential equation models as prediction tools.

In this section we focus on the following question: given the set of data $\{d_j\}_{j=1}^n$ at the respective points $t_j \in [0, T]$, $j = 1, \dots, n$. Find the parameter $a \in Q$ where Q is a compact set contained in $C[0, T]$ (the space of continuous functions on $[0, T]$) which minimizes the least-squares index

$$\sum_{j=1}^n |y(t_j; a) - d_j|^2$$

subject to

$$\frac{dy}{dt} = f(t, y; a), \quad y(0; a) = y_0,$$

where $y(t; a)$ represents the parameter-dependent solution of the above initial-value problem. We combine two methods discussed in this book to provide a numerical algorithm for solving this problem. In particular, we will use approximation theory together with numerical methods for solving differential equations to present an algorithm for solving the least-squares problem. To this end, divide the interval $[0, T]$ into m equal size intervals and denote the bin points by $t_0, t_1, \dots, t_m$. Let φ_i be a spline function (e.g., linear or cubic spline) centered at t_i , $i = 0, \dots, m$ and define $a^m(t) = \sum_{i=0}^m c_i \varphi_i(t)$. Denote by $y_k(a)$ the numerical approximation (using any of the numerical methods discussed in this chapter) of the solution of the differential equation $y(t_k; a)$, $k = 1, \dots, N$ with $t_k - t_{k-1} = h = \frac{T}{N}$. Let $y^N(t; a)$ be a piecewise interpolant (e.g., piecewise linear) of $y_k(a)$ at the points t_k . Then one can define an approximating problem of the above constrained least-squares problem as follows: Find the parameter $a^m \in Q^m$ where Q^m is the space spanned by the $m+1$ spline elements $\varphi_0, \dots, \varphi_m$ which minimizes the least-squares index

$$\sum_{j=1}^n |y^N(t_j; a^m) - d_j|^2.$$

Clearly, the above problem is a finite dimensional minimization problem and is equivalent to the problem: Find $\{c_i\}_{i=1}^m \subset \mathbb{R}^{m+1}$ which minimizes the least-squares index $\sum_{j=1}^n |y^N(t_j; c_0, \dots, c_m) - d_j|^2$. One can apply many optimization routines to solve this problem (e.g., the nonlinear least-squares routine “LSQNONLIN,” available in MATLAB, works well for such a problem).

7.10 Exercises

1. Prove the roundoff error bound (7.24) on page 388.
2. Suppose we consider an example of the initial value problem (7.1) (on page 381), such that $a = 0$, $b = 1$, such that y and f are scalar valued, and such that $f(t, y(t)) = f(t)$, that is, f is a function of the independent variable t only, and not of the dependent variable. In that case,

$$y(1) = y(0) + \int_{t=0}^1 f(t) dt.$$

- (a) To what method of approximating the integral does Euler's method correspond?
 - (b) In view of your answer to item 2a, do you think Euler's method is appropriate to use in practice for accuracy and efficiency?
 - (c) Interpret the total error (roundoff plus truncation) bound (7.24) in this special case. (*One possibility is to take the limit in (7.24) as $L \rightarrow 0$.*)
3. Show that Euler's method fails to approximate the solution $y(x) = (\frac{2}{3}x)^{\frac{3}{2}}$ of the initial value problem $y'(x) = y^{\frac{1}{3}}, y(0) = 0$. Explain why.
 4. Show that $y'(t) = \frac{t}{9} \cos(2y) + t^2, y(0) = 1$, has a unique solution for $|t| \leq 10$.

5. Consider the initial-value problem $\frac{dy}{dt} = a + by(t) + c \sin(y(t)), 0 \leq t \leq 1$ where $y(0) = 1$ and $a, b, c > 0$ are constants. Let us suppose that the solution satisfies $\max_{0 \leq t \leq 1} |y''(t)| = M < \infty$. Consider the approximation $y_{k+1} = y_k + (a + by_k + c \sin(y_k))h$ for $k = 0, 1, 2, \dots, N-1$, $y_0 = y(0)$, and $h = \frac{1}{N}$. Prove that $|y(1) - y_N| \leq \frac{Mhe^{b+c}}{2(b+c)}$.

6. Consider the initial-value problem $\frac{dy}{dt} = f(y, t), y(0) = y_0$, for $0 \leq t \leq 1$. Suppose that $|f(y, t) - f(z, t)| \leq L|y - z|$ for $0 \leq t \leq 1$ and $y, z \in \mathbb{R}$. Also, suppose that the solution $y(t)$ satisfies $\max_{0 \leq t \leq 1} |y''(t)| = M$. Consider the numerical scheme $y_{n+1} = y_n + hf(x_n, t_n) + \epsilon_n$ for $n = 0, 1, 2, \dots, N-1$ where $t_n = nh$, $h = 1/N$ and $y_n \approx y(t_n)$. The ϵ_n are rounding errors and $|\epsilon_n| < \delta$ for all n . Prove that there are constants $c_1, c_2 > 0$ such that, $|y(1) - y_N| < c_1h + c_2\frac{\delta}{h}$.

7. Consider Euler's method for approximating the IVP $y'(x) = f(x, y)$, $0 < x < a$, $y(0) = \alpha$. Let $y_h(x_{i+1}) = y_h(x_i) + hf(x_i, y_h(x_i))$ for $i = 0, 1, \dots, N$ where $y_h(0) = \alpha$. It is known that $y_h(x_i) - y(x_i) = c_1h + c_2h^2 + c_3h^3 + \dots$ where $c_m, m = 1, 2, 3, \dots$ depend on x_i but not on h . Suppose that $y_h(a), y_{\frac{h}{2}}(a), y_{\frac{h}{3}}(a)$ have been calculated using interval width: $h, \frac{h}{2}, \frac{h}{3}$, respectively. Find an approximation $\hat{y}(a)$ to $y(a)$ that is accurate to order h^3 .
8. Compare the midpoint method for IVP's (formula (7.30) on page 390) to the midpoint rule for quadrature (formula (6.28) on page 341).
9. Duplicate the table on page 396, but for $h = 0.05$ and $h = 0.01$. (You will probably want to write a short computer program to do this. You also may need to display more digits than in the original table on page 396.) By taking ratios of errors, illustrate that the global error in the order two Taylor series method is $\mathcal{O}(h^2)$.
10. Suppose that $y'''(t) = t + 2ty'' + 2t^2y(t)$, $1 \leq t \leq 2$, $y(1) = 1$, $y'(1) = 2$, $y''(1) = 3$. Convert this third order equation problem into a first-order system and compute y_k for $k = 1, 2$ for Euler's method with step length $h = 0.1$.
11. Consider the initial-value problem

$$y'(t) = f(t, y), \quad 0 \leq t \leq 1, \quad y(0) = y_0.$$

Suppose

$$\max_{0 \leq t \leq 1} |y'''(t)| = M < \infty, \quad |f(t, u) - f(t, v)| \leq L|u - v|,$$

and

$$\left| \frac{df}{dt}(t, u) - \frac{df}{dt}(t, v) \right| \leq L^2|u - v|, \quad 0 \leq t \leq 1,$$

for constants L and M . Let $h = 1/N$ and

$$y_{j+1} = y_j + hf(t_j, y_j) + \frac{h^2}{2} \frac{df}{dt}(t_j, y_j), \quad j = 0, 1, \dots, N.$$

Prove that $\max_{0 \leq j \leq N} |y_j - y(t_j)| \leq CMh^2$, where C is a constant independent of h .

12. Consider the three stage Runge–Kutta formula

$$\begin{aligned} y_{k+1} &= y_k + h(\gamma_1 K_1 + \gamma_2 K_2 + \gamma_3 K_3), \\ K_1 &= f(t_k, y_k), \\ K_2 &= f(t_k + \alpha_2 h, y_k + h\beta_{21} K_1), \\ K_3 &= f(t_k + \alpha_3 h, y_k + h\beta_{31} K_1 + \beta_{32} K_2). \end{aligned}$$

Determine the set of equations that the coefficients $\{\gamma_j, \alpha_j, \beta_{ji}\}$ must satisfy if the formula is to be of order 3. Find a particular set of values that solves these equations.

13. Calculate the real part of the region for the absolute stability of the fourth order Runge–Kutta method (7.52).
14. Consider the Runge–Kutta method

$$y_{i+1} = y_i + hf(t_i + \frac{h}{8}, y_i + \frac{h}{8}f(t_i, y_i)).$$

Apply this method to $y' = \lambda y$ to find the interval of absolute stability of the method. (Assume that $\lambda h < 0$.)

15. Consider numerical solution of the initial-value problem $y'(t) = f(t, y(t))$, $0 < t < 1$, $y(0) = y_0 = 0$ using the trapezoidal method

$$y_{k+1} = y_k + \frac{h}{2}(f(t_k, y_k) + f(t_{k+1}, y_{k+1})),$$

for $k = 0, 1, \dots, N-1$, where $N = 1/h$ and $t_k = kh$. Suppose that

$$\max_{0 \leq t \leq 1} |y'''(t)| \leq M \quad \text{and} \quad |f(t, z) - f(t, \tilde{z})| \leq L|z - \tilde{z}|$$

for all $z, \tilde{z} \in \mathbb{R}$. Assuming that $hL < 1$, prove that

$$\max_{0 \leq k \leq N} |y_k - y(t_k)| \leq ch^2,$$

where the constant c does not depend on h .

16. Consider the one-step method

$$y_0 = y(t_0), \quad y_{j+1} = y_j + h\Phi(t_j, y_j, h),$$

where

$$\Phi(t_j, y_j, h) = \frac{1}{4}f(t_j, y_j) + \alpha f\left(t_j + \frac{h}{4}, y_j + \beta hf(t_j, y_j)\right).$$

Determine α that will make the method consistent and determine β that will make the method of order of accuracy $p = 1$. Can β be chosen so that the order of accuracy is $p = 2$?

17. Find the region of absolute stability for

(a) Trapezoidal method:

$$y_{j+1} = y_j + \frac{h}{2}(f(t_j, y_j) + f(t_{j+1}, y_{j+1})), \quad j = 0, 1, \dots, N-1.$$

(b) Backward Euler method:

$$y_{j+1} = y_j + hf(t_{j+1}, y_{j+1}), \quad j = 0, 1, \dots, N-1.$$

18. Consider $y'(t) = f(t, y)$ for $0 \leq t \leq 1$, and assume $y \in C^3[0, 1]$. Let

$$\max_{0 \leq t \leq 1} |y''(t)| = k_2 \quad \text{and} \quad \max_{0 \leq t \leq 1} |y'''(t)| = k_3.$$

Let $y_{i+1} = y_i + h\Phi(t_i, y_i, h)$ be a single-step explicit method. Suppose that

$$\left| hy'(t) + \frac{h^2}{2}y''(t) - h\Phi(t, y(t), h) \right| \leq c_1 h^3$$

and

$$|\Phi(t, \tilde{y}, h) - \Phi(t, \hat{y}, h)| \leq M|\tilde{y} - \hat{y}| \quad \text{for} \quad \tilde{y}, \hat{y} \in \mathbb{R}.$$

(a) Prove that $|y(t_i) - y_i| \leq c_2 h^2$ for some constant c_2 , for any $1 \leq i \leq N$.

(b) Prove that $\left| y'(t_i) - \frac{y_{i+1} - y_i}{h} \right| \leq c_3 h$ for some constant c_3 , for any $1 \leq i \leq N$.

19. Consider solving the initial value problem $y' = \lambda y$, $y(0) = \alpha$, where $\lambda < 0$, by the *implicit trapezoid method*, given by

$$y_0 = \alpha, \quad y_{i+1} = y_i + \frac{h}{2} [f(t_{i+1}, y_{i+1}) + f(t_i, y_i)], \quad 0 \leq i \leq N-1,$$

$t_i = ih$, $h = T/N$. Prove that any two numerical solutions y_i and $\hat{y}_i$ satisfy

$$|y_i - \hat{y}_i| \leq e^K |y_0 - \hat{y}_0|$$

for $0 \leq t_i \leq T$, assuming that $\lambda h \leq 1$, where $K = 3\lambda T/2$ and $y_0, \hat{y}_0$ are respective initial values with $y_0 \neq \hat{y}_0$. (That is, y_i and $\hat{y}_i$ satisfy the same difference equations except for different initial values.)

20. Consider the initial-value system

$$\frac{dy}{dt} = (I - Bt)^{-1}y, \quad y(0) = y_0, \quad y(t) \in \mathbb{R}^n, \quad 0 \leq t \leq 1,$$

where B is an $n \times n$ matrix with $\|B\|_\infty \leq 1/2$. Euler's method for approximating $y(t)$ has the form

$$y_{i+1} = y_i + h(I - Bt_i)^{-1}y_i = (I + h(I - Bt_i)^{-1})y_i, \quad i = 0, 1, \dots, N-1,$$

where $t_i = ih$ and $h = 1/N$. Noting that $\|Bt_i\|_\infty \leq 1/2$ for all i , prove that

$$\|y_{i+1}\|_\infty \leq (1 + 2h)\|y_i\|_\infty$$

for $i = 0, 1, \dots, N - 1$ and

$$\|y_N\|_\infty \leq e^2 \|y_0\|_\infty$$

for any value of $N \geq 1$.

21. Show that the implicit Simpson's method (defined by Equation (7.67) on page 405) has order of accuracy 4.
22. Consider the following two-step method:

$$y_{k+2} + \alpha_1 y_{k+1} + \alpha_0 y_k = h\beta f(t_{k+2}, y_{k+2})$$

for solving the initial-value problem $y'(t) = f(t, y)$.

- (a) Find $\alpha_0, \alpha_1, \beta$ such that the method is *second* order.
 - (b) Is the method consistent? Is so why and if not why not?
 - (c) Is the method stable? Is so why and if not why not?
23. Consider the initial value problem

$$\frac{dy}{dt} = f(t, y), \quad y(a) = \eta$$

for the function $y(t)$ over the interval $a \leq t \leq b$. Consider the general multistep method on the discrete point set defined by $x_n = a + nh$ for $n = 0, \dots, m$ with $h = (b-a)/m$. If we write $y_n = y(t_n)$ and $f_n = f(t_n, y_n)$ the general multistep method takes the form

$$\sum_{j=0}^k \alpha_j y_{n+j} = h \sum_{j=0}^k \beta_j f_{n+j}.$$

- (a) Assuming $\alpha_k = 1$, construct the implicit linear two step method containing one free parameter $\alpha_0 = c$.
 - (b) Find the order of this two-step method as a function of the parameter c . Determine the value of c in which this two-step method has maximal order.
 - (c) Find a value of the parameter c for which the method is explicit.
24. Consider the problem $y'(t) = t$, $y(0) = 0$ with the exact solution $y(t) = t^2/2$. Consider the three methods:
 - (i) $y_{j+2} + y_{j+1} - 2y_j = 2hf(t_j, y_j)$ with $y_0 = y(0) = 0$, $y_1 = y(h) = h^2/2$,
 - (ii) $y_{j+1} - y_j = 2hf(t_j, y_j)$ with $y_0 = y(0) = 0$,
 - (iii) $y_{j+1} - y_j = 2hf(t_{j+1}, y_{j+1})$ with $y_0 = y(0) = 0$,

where $t_j = jh$. Show that method (i) is consistent but not stable, method (ii) is stable but not consistent, and method (iii) is stable and consistent.

25. Determine the value of c that will make the multistep method

$$y_{j+2} = -4y_{j+1} + 5y_j + chf_{j+1} + 2hf_j$$

consistent.

26. Determine whether or not the method in Exercise 25 is stable.

27. Consider the predictor-corrector pair

$$\begin{cases} y_{k+1}^{(0)} = y_k + hf(t_k, y_k), \\ y_{k+1} = y_k + \alpha hf(t_k, y_{k+1}^{(0)}) + (1 - \alpha)hf(t_k, y_k), \end{cases}$$

where $0 < \alpha < 1$ is a parameter. Suppose that $\alpha = \frac{1}{3}$. Find the interval of absolute stability of the resulting method.

28. Develop a general-purpose routine for solving the IVP $y' = f(t, y)$, $y(t_0) = \alpha$, using the following Adams predictor-corrector algorithm.

ALGORITHM 7.1

(A predictor-corrector scheme)

INPUT: the end points a and b , the initial condition α , and the number of subintervals N .

OUTPUT: approximate values $\{t_i, y_i\}_{i=0}^N$ of the solution y at the points $t_i = a + \frac{b-a}{N}i$.

1. Given $y_0 = \alpha$, use the fourth-order Runge-Kutta method to determine y_1, y_2, y_3 .
2. DO for $n = 3, \dots, N - 1$
 - (a) Use the four-step Adams-Bashforth Method for $i = 3, 4, \dots, N - 1$, as predictor to compute the approximation y_{n+1}^P using the values $y_n, y_{n-1}, y_{n-2}, y_{n-3}$.
 - (b) Use the three-step Adams-Moulton Method, for $i = 2, 3, \dots, N - 1$, as corrector to compute the approximation y_{n+1}^C using the values $y_{n+1}^P, y_n, y_{n-1}, y_{n-2}$.
 - (c) $y_{n+1} \leftarrow y_{n+1}^C$.

END DO

END ALGORITHM 7.1.

29. Consider solving the IVP $y' = t + y$, $0 \leq t \leq 1$, $y(0) = 0$ using the program developed in Exercise 28 with $N = 10, 20, 40, 80, 160, 320, 640$.
- Corresponding to each N , let the approximate solution computed at the right end point be denoted by $y_1, y_2, y_3, y_4, y_5, y_6, y_7$.
 - Compute the exact solution to this problem evaluated at the right end point, and denote it by y_e .
 - Compute the error $e_i = |y_i - y_e|$, $i = 1, \dots, 7$ corresponding to each N .
 - Compute the ratios $\alpha_i = e_{i+1}/e_i$, $i = 1, \dots, 6$.

What does α_i converge to and why?

30. Develop a general purpose routine to solve systems of differential equations by suitably modifying Algorithm 7.1. Use this new program with $N = 10$, to do the following:
- Solve the second-order IVP $y'' + 3y' + 2y = 6e^t$ on $0 \leq t \leq 1$, given $y(0) = 3$ and $y'(0) = -2$. Determine the exact solution to this problem. Plot the exact solution and the computed solution over $0 \leq t \leq 1$ and verify that the computed solution compares favorably with the exact solution.
 - Express the system of equations

$$\begin{aligned}\frac{d^2 z}{dt^2} &= z^2 - y + e^t \\ \frac{d^2 y}{dt^2} &= z - y^2 - e^t\end{aligned}$$

with the initial conditions,

$$z(0) = z'(0) = 0, \quad y(0) = 1, \quad y'(0) = -2$$

as a system of first-order equations and solve it from $t = 0$ to $t = 1$.

31. Show that, when A has simple eigenvalues, the solution to the linear initial value problem (7.95) on page 418 is (7.96). *Hint: Letting P be the matrix whose columns are the n linearly independent eigenvectors of A , make the transformation $y = Pz$, and solve the resulting linear system in z .*
32. Show that the method presented in Example 7.11 on page 426 is A-stable.
33. Show that the method presented in Example 7.11 on page 426 is of order $\mathcal{O}(h^3)$.

34. Consider the following time-dependent logistic model for $t \in [0, 2]$:

$$\frac{dy}{dt} = a(t)y\left(1 - \frac{y}{5}\right), \quad y(0) = 4.$$

- (a) Find parameters c_i to approximate the time-varying coefficient $a(t) \approx \sum_{i=0}^2 c_i \varphi_i(t)$. Here, φ_i denotes the hat function centered at t_i , with respect to the nodes $[t_0, t_1, t_2] = [0, 1, 2]$. (See page 239.) Compute those c_i which provide the best least-squares fit for the (t, a) data set:

$$\{(0.3, 5), (0.6, 5.2), (0.9, 4.8), (1.2, 4.7), (1.5, 5.5), (1.8, 5.2), (2, 4.9)\}.$$

- (b) Solve the resulting initial value problem numerically. Somehow estimate the error in your numerical solution.
35. Show that the exact solutions of the difference equations (i), (ii), and (iii) of Exercise 24 are given respectively by:

$$(i) \quad y_j = -\frac{5}{18}h^2(-2)^j + \frac{5}{18}h^2 - \frac{5}{6}h^2j + \frac{h^2}{2}j^2,$$

$$(ii) \quad y_j = -h^2j + h^2j^2, \text{ and}$$

$$(iii) \quad h_j = \frac{h^2}{2}j + \frac{h^2}{2}j^2.$$

That is, show that these solutions satisfy the difference equations and the initial conditions.

Now, compare the exact solution $y(t_j) = j^2h^2/2$ to these solutions. In particular, consider $h = 0.1$, $j = 10$, and $h = 0.01$, $j = 100$ for estimating $y(1) = 1/2$.

Chapter 8

Numerical Solution of Systems of Nonlinear Equations

In this chapter, we study numerical methods for solution of nonlinear systems. Two classic references on numerical solution of nonlinear systems are [70] and [73].

We are interested in finding $x = (x_1, x_2, \dots, x_n)^T \in D \subset \mathbb{R}^n$ that solves

$$F(x) = 0, \quad (8.1)$$

where $F(x) = (f_1(x), f_2(x), f_3(x), \dots, f_n(x))^T$, $F : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$. For example,

$$F(x) = \begin{pmatrix} f_1(x_1, x_2, x_3) \\ f_2(x_1, x_2, x_3) \\ f_3(x_1, x_2, x_3) \end{pmatrix} = \begin{pmatrix} e^{x_1 x_2} + x_1 + 5x_3 x_1^3 \\ 1 + 3x_1 + 4x_2^2 + x_3^5 x_1 \\ 4 + x_1 - 2x_2 + 4x_3 \end{pmatrix}.$$

Here, we consider iterative methods of solution of these nonlinear systems, the most general techniques.¹

8.1 Introduction and Fréchet Derivatives

In this section, we review Fréchet derivatives, a useful concept throughout this chapter.

DEFINITION 8.1 *A mapping $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ is Fréchet- or F -differentiable at an interior point x of D if there exists a linear mapping*

¹Symbolic algebra methods have become more popular in recent years, and are sometimes combined with iterative methods, as a way to preprocess systems. Also, exhaustive search methods, such as explained in §9.6.3 of this book, can sometimes be used for small systems to find *all* solutions. Continuation methods are sometimes also used to find all solutions. However, the iterative methods of this chapter remain a basic element of most software for finding solutions, and are the primary element when the system is large.

$A : \mathbb{R}^n \rightarrow \mathbb{R}^n$ such that for any $h \in \mathbb{R}^n$,

$$\lim_{h \rightarrow 0} \frac{1}{\|h\|} \|F(x+h) - F(x) - Ah\| = 0, \quad (8.2)$$

where $\|\cdot\|$ is a vector norm in $\mathbb{R}^n$.

REMARK 8.1 A is an $n \times n$ matrix dependent on point x , i.e., $A = A(x)$.
□

DEFINITION 8.2 The linear map A for which (8.2) holds is called the Fréchet derivative of F at x , and is denoted $F'(x)$.

Now, suppose that

$$F(x) = (f_1(x), f_2(x), f_3(x), \dots, f_n(x))^T,$$

where $f_j(x)$ has continuous first partial derivatives on D . Let $(F'(x))_{ij} = a_{ij}(x)$ be the (i, j) -th element of A . Since convergence in norm implies componentwise convergence, (8.2) with $h = te_j$, where $e_j = (0, \dots, 0, 1, 0, \dots, 0)^T$ is the j -th unit vector, gives

$$\lim_{t \rightarrow 0} \frac{1}{t} |f_i(x + te_j) - f_i(x) - ta_{ij}(x)| = 0 \quad \text{for } 1 \leq i \leq n. \quad (8.3)$$

However,

$$\lim_{t \rightarrow 0} \frac{1}{t} (f_i(x + te_j) - f_i(x)) = \frac{\partial f_i(x)}{\partial x_j} \quad (\text{evaluated at } x).$$

Hence, (8.3) implies that $(F'(x))_{ij} = a_{ij}(x) = \frac{\partial f_i(x)}{\partial x_j}$, $1 \leq i \leq n$, $1 \leq j \leq n$.

Thus,

$$F'(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & \frac{\partial f_1}{\partial x_2}(x) & \dots & \frac{\partial f_1}{\partial x_n}(x) \\ \frac{\partial f_2}{\partial x_1}(x) & \frac{\partial f_2}{\partial x_2}(x) & \dots & \frac{\partial f_2}{\partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}(x) & \frac{\partial f_n}{\partial x_2}(x) & \dots & \frac{\partial f_n}{\partial x_n}(x) \end{pmatrix}. \quad (8.4)$$

DEFINITION 8.3 The matrix of partial derivatives in Equation (8.4) is called the Jacobian matrix for the function F .

Example 8.1

If

$$F(x) = \begin{pmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{pmatrix} = \begin{pmatrix} x_1 \cos x_2 + 3 + e^{x_2^2} \\ x_1 x_2^2 - x_1 + 2x_2 \end{pmatrix},$$

then

$$F'(x) = \left(\begin{array}{c|c} \cos x_2 & -x_1 \sin x_2 + 2x_2 e^{x_2^2} \\ \hline x_2^2 - 1 & 2x_1 x_2 + 2 \end{array} \right).$$

□

Mean-value theorems as well as generalized Taylor series in many dimensions can be derived using Fréchet derivatives. A thorough theoretical treatment is presented in [104]. For example, a mean value theorem for a function $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$ can be stated as

THEOREM 8.1

(A multivariate mean value theorem) Suppose $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ has continuous first-order partial derivatives, and suppose that $x \in D$, $\tilde{x} \in D$, and the line segment $\{\tilde{x} + t(x - \tilde{x}) \mid t \in [0, 1]\}$ is in D . Then

$$F(x) = F(\tilde{x}) + A(x - \tilde{x}), \quad (8.5)$$

where A is some matrix whose i -th row is of the form

$$\left(\frac{\partial f_i}{\partial x_1}(c_i), \frac{\partial f_i}{\partial x_2}(c_i), \dots, \frac{\partial f_i}{\partial x_n}(c_i) \right),$$

where the $c_i \in \mathbb{R}^n$, $1 \leq i \leq n$ are (possibly distinct) points on the line between $\tilde{x}$ and x .

PROOF The proof is Exercise 1 below.

□

Higher-order Taylor expansions are of the form

$$F(x) = F(\tilde{x}) + F'(\tilde{x})(x - \tilde{x}) + \frac{1}{2}F''(\tilde{x})(x - \tilde{x})(x - \tilde{x}) + \dots, \quad (8.6)$$

where F' is the Jacobian matrix as in (8.4) and F'' , F''' , etc. are *higher-order derivative tensors*. For example, $F''(x)$ can be viewed as a matrix of matrices, whose (i, j, k) -th element is

$$\frac{\partial^2 f_i}{\partial x_j \partial x_k}(x),$$

and where $F''(\tilde{x})(x - \tilde{x})$ can be viewed as a matrix whose (i, j) -th entry is computed as

$$(F''(\tilde{x})(x - \tilde{x}))_{i,j} = \sum_{k=1}^n \frac{\partial^2 f_i}{\partial x_j \partial x_k}(\tilde{x})(x_k - \tilde{x}_k).$$

Just as in univariate Taylor expansions, if we truncate the expansion in Equation 8.6 by taking terms only up to and including the k -th Fréchet derivative, then the resulting multivariate Taylor polynomial $T_k(x)$ satisfies

$$F(x) = T_k(x) + \mathcal{O}(\|x - \tilde{x}\|^{k+1}).$$

8.2 Successive Approximation (Fixed Point Iteration) and the Contraction Mapping Theorem

We now consider the successive approximation method. This is a close multidimensional analogue to the univariate fixed point iteration method discussed in Section 2.3 on page 39. In turn, there are also infinite-dimensional analogues to this development that are useful in the analysis of differential equations.

Suppose that a solution of $F(x) = 0$ is a *fixed point* of G , i.e., $x = G(x)$. We have the iteration scheme

$$x^{(k+1)} = G(x^{(k)}), \quad k \geq 0, \quad x^{(0)} \text{ given in } \mathbb{R}^n, \quad (8.7)$$

where

$$G(x^{(k)}) = (g_1(x^{(k)}), g_2(x^{(k)}), \dots, g_n(x^{(k)}))^T. \quad (8.8)$$

DEFINITION 8.4 Let $G : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$. Then x^* is a point of attraction of iteration (8.7) if there is an open neighborhood S of x^* such that $S \subset D$ and, for any $x^{(0)} \in S$, the iterations $\{x^{(k)}\}$ all lie in D and converge to x^* .

We now begin with the contraction mapping theorem. First,

DEFINITION 8.5 A mapping $G : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a contraction on a set $D_0 \subset D$ if there is an $\alpha < 1$ such that $\|G(x) - G(y)\| \leq \alpha \|x - y\|$ for all $x, y \in D_0$.

THEOREM 8.2

(Contraction Mapping Theorem) Suppose that $G : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a contraction on a closed set $D_0 \subset D$ and that $G : D_0 \rightarrow D_0$, i.e., if $x \in D_0$, then $G(x) \in D_0$. Then G has a unique fixed point $x^* \in D_0$. Moreover, for any $x^{(0)} \in D_0$, the iterates $\{x^{(k)}\}$ defined by $x^{(k+1)} = G(x^{(k)})$ converge to x^* . We also have the error estimates

$$\|x^{(k)} - x^*\| \leq \frac{\alpha}{1 - \alpha} \|x^{(k)} - x^{(k-1)}\|, \quad k = 1, 2, \dots \quad (8.9)$$

$$\|x^{(k)} - x^*\| \leq \frac{\alpha^k}{1 - \alpha} \|G(x^{(0)}) - x^{(0)}\|, \quad (8.10)$$

where α is as in Definition 8.5.

PROOF (This is similar to the proof of Theorem 2.3 on page 40, the univariate contraction mapping theorem.) Let $x^{(0)}$ be an arbitrary point in D_0 . Since $G(D_0) \subset D_0$ (where $G(D_0)$ is the set $\{G(x) \mid x \in D_0\}$), the sequence defined by $x^{(k+1)} = G(x^{(k)})$ is well defined and lies in D_0 . By Definition 8.5,

$$\|x^{(k+1)} - x^{(k)}\| = \|G(x^{(k)}) - G(x^{(k-1)})\| \leq \alpha \|x^{(k)} - x^{(k-1)}\| \quad \text{for } k \geq 1. \quad (8.11)$$

Repeated application of (8.11) yields

$$\begin{aligned} \|x^{(k+p)} - x^{(k)}\| &\leq \sum_{i=1}^p \|x^{(k+i)} - x^{(k+i-1)}\| \\ &\leq (\alpha^{p-1} + \alpha^{p-2} + \cdots + 1) \|x^{(k+1)} - x^{(k)}\| \\ &\leq \frac{1}{1 - \alpha} \|x^{(k+1)} - x^{(k)}\| \leq \frac{\alpha^k}{1 - \alpha} \|x^{(1)} - x^{(0)}\|. \end{aligned} \quad (8.12)$$

Thus, by (8.12) $\{x^{(k)}\}$ is a Cauchy sequence (that is, there is an n_0 such that $\|f_m - f_n\| < \epsilon$ whenever $m, n > n_0$) in the closed set $D_0 \subset \mathbb{R}^n$, with respect to any norm. Therefore, since Cauchy sequences in $\mathbb{R}^n$ converge to elements in $\mathbb{R}^n$,

$$\lim_{k \rightarrow \infty} x^{(k)} = x^* \quad \text{and } x^* \in D_0. \quad (8.13)$$

In addition,

$$\begin{aligned} \|x^* - G(x^*)\| &= \|x^* - x^{(k+1)} + G(x^k) - G(x^*)\| \\ &\leq \|x^* - x^{(k+1)}\| + \|G(x^k) - G(x^*)\| \\ &\leq \|x^* - x^{(k+1)}\| + \alpha \|x^k - x^*\| \\ &\rightarrow 0 \quad \text{as } k \rightarrow \infty. \end{aligned}$$

Thus, since G is continuous, $x^* = G(x^*)$, i.e., x^* is a fixed point of G . Furthermore, the limit is unique, since if there were two fixed points $x_1^* \neq x_2^*$ in D_0 , then $\|x_1^* - x_2^*\| = \|G(x_1^*) - G(x_2^*)\| \leq \alpha \|x_1^* - x_2^*\| < \|x_1^* - x_2^*\|$, which is a contradiction. Finally, error estimates (8.9) and (8.10) follow from (8.12) by letting $p \rightarrow \infty$ and observing that $x^{(1)} = G(x^{(0)})$. $\square$

Example 8.2

Consider

$$\begin{aligned} 3x_1 - \cos x_1 x_2 - \frac{1}{2} &= 0, \quad \text{or } x_1 = \frac{1}{3} \cos x_1 x_2 + \frac{1}{6} \\ 20x_2 + e^{-x_1 x_2} + 8 &= 0, \quad \text{or } x_2 = -\frac{1}{20} e^{-x_1 x_2} - \frac{8}{20}. \end{aligned}$$

Thus, the related systems $F(x) = 0$ and $G(x) = x$ can be written

$$F(x) = (3x_1 - \cos x_1 x_2 - \frac{1}{2}, 20x_2 + e^{-x_1 x_2} + 8)^T,$$

and

$$G(x) = (g_1(x), g_2(x))^T = \left(\frac{1}{3} \cos x_1 x_2 + \frac{1}{6}, -\frac{1}{20} e^{-x_1 x_2} - \frac{8}{20} \right)^T.$$

Let $D_0 = \{(x_1, x_2) \in \mathbb{R}^2 : -1 \leq x_i \leq 1 \text{ for } i = 1, 2\}$. We will use Theorem 8.2 to show that the iterations $x^{(k+1)} = G(x^{(k)})$ converge to a unique fixed point $x^* \in D_0$. Consider

$$\begin{aligned} |g_1(x_1, x_2)| &= \left| \frac{1}{3} \cos x_1 x_2 + \frac{1}{6} \right| \\ &\leq \frac{1}{6} + \frac{1}{3} |\cos x_1 x_2| \leq \frac{1}{2} \\ |g_2(x_1, x_2)| &= \left| \frac{1}{20} e^{-x_1 x_2} + \frac{2}{5} \right| \\ &\leq \frac{2}{5} + \frac{1}{20} e \leq .55 \quad \text{for } -1 \leq x_1, x_2 \leq 1. \end{aligned}$$

Thus, $|g_i(x)| \leq 1$ for $i = 1, 2$ and hence $-1 \leq g_i(x) \leq 1$ for $i = 1, 2$ if $x \in D_0$. (In other words, $G(x) \in D_0$ whenever $x \in D_0$.) Now consider showing G is a contraction on D_0 . We need to show that there is an $\alpha < 1$ such that $\|G(x) - G(y)\| \leq \alpha \|x - y\|$ for all $x, y \in D_0$. $\square$

In Example 8.2, showing that G is a contraction can be facilitated by Theorem 8.3 below. To present Theorem 8.3, we first review the following definition.

DEFINITION 8.6 A set D_0 is said to be convex, provided

$$\lambda x + (1 - \lambda)y \in D_0 \quad \text{whenever } x \in D_0, y \in D_0, \text{ and } \lambda \in [0, 1].$$

THEOREM 8.3

Let D_0 be a convex subset of $\mathbb{R}^n$ and G be a mapping of D_0 into $\mathbb{R}^n$ whose components $g_1, g_2, \dots, g_n$ have continuous and bounded derivatives of first order on D_0 . Then the mapping G satisfies the Lipschitz condition

$$\|G(x) - G(y)\| \leq L \|x - y\| \quad \text{for all } x, y \in D_0, \quad (8.14)$$

where $L = \sup_{w \in D_0} \|G'(w)\|$, where G' is the Jacobian matrix of G (i.e., G' is the Fréchet derivative of G). If $L \leq \alpha < 1$, then G is a contraction on D_0 .

REMARK 8.2 $\|\cdot\|$ signifies a vector norm and the corresponding induced matrix norm, i.e., $\|A\| = \sup_{\|x\| \neq 0} \frac{\|Ax\|}{\|x\|}$. Thus, $\|Ax\| \leq \|A\|\|x\|$. $\square$

PROOF Since D_0 is convex, $x + s(y - x) \in D_0$ for $0 \leq s \leq 1$ for any $y, x \in D_0$. Let

$$\Phi_j(s) = g_j(x + s(y - x)), \quad 0 \leq s \leq 1, \quad j = 1, 2, \dots, n. \quad (8.15)$$

Observe that Φ_j is a continuously differentiable function of s on $[0, 1]$, because

$$\frac{d\Phi_j}{ds} = \frac{\partial g_j}{\partial x_1} \frac{\partial x_1}{\partial s} + \dots + \frac{\partial g_j}{\partial x_n} \frac{\partial x_n}{\partial s} = \frac{\partial g_j}{\partial x_1}(y_1 - x_1) + \dots + \frac{\partial g_j}{\partial x_n}(y_n - x_n).$$

With these functions Φ_j ,

$$g_j(y) - g_j(x) = \Phi_j(1) - \Phi_j(0) = \int_0^1 \Phi'_j(s) ds \quad (8.16)$$

and

$$\Phi'_j(s) = \sum_{k=1}^n \frac{\partial g_j(w)}{\partial x_k} (y_k - x_k), \quad (8.17)$$

where $w = x + s(y - x)$.

Let $\Phi(s) = (\Phi_1(s), \Phi_2(s), \dots, \Phi_n(s))^T$. Then (8.16) and (8.17) give

$$G(y) - G(x) = \int_0^1 \Phi'(s) ds \quad (8.18)$$

and

$$\Phi'(s) = G'(w)(y - x), \quad (8.19)$$

where $G'(w)$ is the Jacobian matrix of G evaluated at $w = x + s(y - x)$.

Thus,

$$\|G(y) - G(x)\| = \left\| \int_0^1 \Phi'(s) ds \right\| \leq \sup_{0 \leq s \leq 1} \|\Phi'(s)\|. \quad (8.20)$$

To see the above inequality, note that

$$\sum_{\nu=1}^N \Phi' \left(\frac{\nu}{N} \right) \frac{1}{N} \rightarrow \int_0^1 \Phi'(s) ds \quad \text{as } N \rightarrow \infty.$$

But

$$\left\| \sum_{\nu=1}^N \Phi' \left(\frac{\nu}{N} \right) \frac{1}{N} \right\| \leq \sum_{\nu=1}^N \left\| \Phi' \left(\frac{\nu}{N} \right) \right\| \frac{1}{N} \leq \sup_{0 \leq s \leq 1} \|\Phi'(s)\|.$$

Hence,

$$\left\| \int_0^1 \Phi'_j(s) ds \right\| = \lim_{N \rightarrow \infty} \left\| \sum_{\nu=1}^N \Phi' \left(\frac{\nu}{N} \right) \frac{1}{N} \right\| \leq \sup_{0 \leq s \leq 1} \|\Phi'(s)\|.$$

Inequality (8.20) and Equation (8.19) now give

$$\|G(y) - G(x)\| \leq \sup_{0 \leq s \leq 1} \|\Phi'(s)\| \leq \sup_{0 \leq s \leq 1} \|G'(w)\| \|y - x\|.$$

Hence,

$$\|G(y) - G(x)\| \leq L\|y - x\|, \quad \text{where } L = \sup \|G'(w)\|.$$

□

Now let's apply this theorem to Example 8.2. We have

$$G'(x) = \begin{pmatrix} \frac{dg_1}{dx_1} & \frac{dg_1}{dx_2} \\ \frac{dg_2}{dx_1} & \frac{dg_2}{dx_2} \end{pmatrix} = \begin{pmatrix} \frac{1}{3}x_2 \sin(x_1 x_2) & \frac{1}{3}x_1 \sin(x_1 x_2) \\ \frac{x_2}{20}e^{-x_1 x_2} & \frac{x_1}{20}e^{-x_1 x_2} \end{pmatrix}$$

and

$$\left| \frac{dg_1}{dx_1} \right| \leq \frac{1}{3}, \quad \left| \frac{dg_1}{dx_2} \right| \leq \frac{1}{3}, \quad \left| \frac{dg_2}{dx_1} \right| \leq \frac{e}{20}, \quad \text{and} \quad \left| \frac{dg_2}{dx_2} \right| \leq \frac{e}{20}$$

for $x \in D_0 = [-1, 1] \times [-1, 1]$. Thus,

$$\|G'(x)\|_\infty \leq \frac{2}{3} = L.$$

Hence, G is a contraction on D_0 , $G(D_0) \subset D_0$, and Theorem 8.2 implies that G has a unique fixed point in D_0 and the iterations defined by $x^{(k+1)} = G(x^{(k)})$ converge to this unique fixed point.

We also have

THEOREM 8.4

Let x^* be a fixed point of $G(x)$, and assume the components of $G(x)$ are continuously differentiable in some neighborhood of x^* . Furthermore, assume that $\|G'(x^*)\| < 1$, where $\|\cdot\|$ is some vector norm and corresponding induced matrix norm. Then, for x_0 chosen sufficiently close to x^* , $x^{(k+1)} = G(x^{(k)})$ will converge to x^* , and the results of Theorem 8.2 will be valid on some closed, bounded, convex region about x^* .

PROOF Pick a number λ satisfying $\|G'(x^*)\| < \lambda < 1$. Then, choose a set

$$D_0 = \{x \in \mathbb{R}^n : \|x - x^*\| \leq \epsilon\},$$

with $L = \max_{x \in D_0} \|G'(x)\| \leq \lambda < 1$. We have $G(D_0) \subset D_0$, since $\|x^* - x\| \leq \epsilon$ implies that

$$\|x^* - G(x)\| = \|G(x^*) - G(x)\| \leq L\|x^* - x\| \leq \lambda\|x^* - x\| < \epsilon.$$

Thus, G maps D_0 into D_0 and is a contraction on D_0 , so Theorem 8.2 can be applied. $\square$

8.3 Newton's Method and Variations

We specialize the results of the previous section to iterations of the form

$$x^{(k+1)} = x^{(k)} - (A(x^{(k)}))^{-1}F(x^{(k)}), \quad (8.21)$$

i.e.,

$$G(x) = x - A^{-1}(x)F(x),$$

where $A(x)$ is an $n \times n$ matrix whose elements are functions of x . Assuming $A(x)$ is nonsingular, $x = G(x)$ (i.e., x is a fixed point of G) if and only if $F(x) = 0$.

REMARK 8.3 Equation (8.21) is equivalent to

$$x^{(k+1)} = x^{(k)} + v^{(k)}, \quad \text{where } v^{(k)} \text{ solves } A(x^{(k)})v^{(k)} = -F(x^{(k)}), \quad (8.22)$$

where (8.22) is how the iteration is implemented in practice. $\square$

REMARK 8.4 If $A(x) = F'(x)$, then (8.21) is called *Newton's method*. $\square$

8.3.1 Some Local Convergence Results for Newton's Method

We will derive a local convergence result for Newton's method. First, we consider several preliminary results.

LEMMA 8.1

Let A be a nonsingular matrix. Suppose that B is a matrix such that $\|A^{-1}\|\|B\| < 1$. Then, $A + B$ is nonsingular, and

$$\|(A + B)^{-1}\| \leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\|\|B\|}. \quad (8.23)$$

PROOF $A + B = A(I + A^{-1}B)$, but $\|A^{-1}B\| \leq \|A^{-1}\|\|B\| < 1$. Thus, $\rho(A^{-1}B) < 1$, so $A(I + A^{-1}B)$ is nonsingular, from which it follows that $A + B$ is nonsingular. Also, $(A + B)^{-1} = (I + A^{-1}B)^{-1}A^{-1}$, so

$$\|(A + B)^{-1}\| \leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\|\|B\|}.$$

(Recall if $\|C\| < 1$, then $(I - C)^{-1} = I + C + C^2 + \cdots$, so

$$\|(I - C)^{-1}\| \leq 1 + \|C\| + \|C\|^2 + \cdots = \frac{1}{1 - \|C\|}.)$$

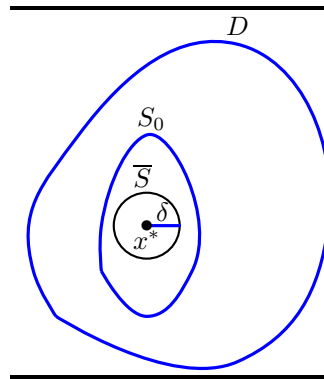
□

THEOREM 8.5

Suppose that $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ is F -differentiable at a point x^* in the interior of D at which $F(x^*) = 0$. Let $A : S_0 \rightarrow \mathbb{R}^n$ be a linear mapping (i.e., A is an $n \times n$ matrix), let A be continuous at x^* , and let $A(x^*)$ be nonsingular. Then, there exists a closed ball $\bar{S}(x^*, \delta) \subset S_0$, $\bar{S} = \{x \in S_0 \subset D : \|x - x^*\| \leq \delta\}$, $\delta > 0$, on which the mapping $G : \bar{S} \rightarrow \mathbb{R}^n$, $G(x) = x - A^{-1}(x)F(x)$ is well-defined. Moreover, G is F -differentiable at x^* and

$$G'(x^*) = I - (A(x^*))^{-1}F'(x^*).$$

(See the following figure.)



PROOF G will be well-defined on $\bar{S}$ if A is nonsingular on $\bar{S}$. Let $\beta = \|A^{-1}(x^*)\|$ and let $\epsilon > 0$ be such that $0 < \epsilon < \frac{1}{2\beta}$. By hypothesis, $A(x)$ is continuous at x^* . Therefore, there is a $\delta = \delta(x^*, \epsilon) > 0$ such that $\bar{S}(x, \delta) \subset \bar{S}_0$ and

$$\|A(x) - A(x^*)\| \leq \epsilon \quad \text{for all } x \in \bar{S}. \quad (8.24)$$

We now use Lemma 8.1 with $A = A(x)$ and $B = -A(x^*) + A(x)$, i.e., $A + B = A(x)$. Since

$$\|A^{-1}(x^*)\| \|A(x) - A(x^*)\| \leq \epsilon\beta < \frac{1}{2} \quad \text{for all } x \in \bar{S},$$

$A(x)$ is invertible for all $x \in \overline{S}$. Moreover,

$$\begin{aligned} \|A^{-1}(x)\| &= \|(A(x^*) + A(x) - A(x^*))^{-1}\| \\ &\leq \frac{\|A^{-1}(x^*)\|}{1 - \|A^{-1}(x^*)\|\|A(x) - A(x^*)\|} \quad (\text{by Lemma 8.1}) \\ &\leq \frac{\beta}{1 - \frac{1}{2}} = 2\beta \quad \text{for } x \in \overline{S}. \end{aligned} \quad (8.25)$$

Thus, G is well-defined for all $\bar{x} \in \overline{S}$. We now show that G is F-differentiable at x^* . First, since x^* is a solution of $F(x) = 0$, x^* is a fixed point of G , i.e.,

$$G(x^*) = x^*. \quad (8.26)$$

Also, since F is F-differentiable at x^* and by choosing δ , the radius of $\overline{S}(x^*, \delta)$, sufficiently small, and since

$$\lim_{\|x - x^*\| \rightarrow 0} \frac{\|F(x) - F(x^*) - F'(x^*)(x - x^*)\|}{\|x - x^*\|} = 0,$$

we conclude that

$$\|F(x) - F(x^*) - F'(x^*)(x - x^*)\| \leq \epsilon \|x - x^*\| \quad \text{for all } x \in \overline{S}. \quad (8.27)$$

Now, for $x \in \overline{S}$,

$$\begin{aligned} &\|G(x) - G(x^*) - (I - A^{-1}(x^*)F'(x^*))(x - x^*)\| \\ &= \|A^{-1}(x^*)F'(x^*)(x - x^*) - A^{-1}(x)F(x)\| \\ &\leq \|A^{-1}(x)[F(x) - F(x^*) - F'(x^*)(x - x^*)]\| \\ &\quad + \|A^{-1}(x)(A(x^*) - A(x))A^{-1}(x^*)F'(x^*)(x - x^*)\| \quad \text{since } F(x^*) = 0 \\ &\leq 2\beta\epsilon\|x - x^*\| + 4\beta^2\epsilon\|F'(x^*)\|\|x - x^*\| \quad \text{using (8.24), (8.25), and (8.27)} \\ &\leq \epsilon C\|x - x^*\|, \quad \text{where } C = 2\beta + 4\beta^2\|F'(x^*)\| \text{ is a constant.} \end{aligned}$$

The above computation, combined with the definition of Fréchet derivative (Definition 8.1), shows that G is F-differentiable at x^* and $G'(x^*) = I - A^{-1}(x^*)F'(x^*)$. $\square$

Before proving a local convergence theorem for iteration (8.21), we introduce the following lemma. A proof of this lemma can be found, for example, in [70].

LEMMA 8.2

(Ostrowski) Assume that $G : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ has a fixed point x^* in the interior of D and that G has a Fréchet derivative G' at x^* . Then, if the spectral radius of $G'(x^*)$ satisfies $\rho(G'(x^*)) = \sigma < 1$, it follows that x^* is a point of attraction of the iterates $x^{(k+1)} = G(x^{(k)})$.

We now have an attraction result for iteration (8.21).

COROLLARY 8.1

(to Theorem 8.5) Assume that the hypotheses of Theorem 8.5 hold. In addition, suppose that

$$\rho(G'(x^*)) = \rho(I - A^{-1}(x^*)F'(x^*)) = \sigma < 1. \quad (8.28)$$

Then x^* is a point of attraction of the iterations 8.21.

PROOF Lemma 8.2 and Theorem 8.5 imply that x^* is a point of attraction of iteration

$$x^{(k+1)} = x^{(k)} - A^{-1}(x^{(k)})F(x^{(k)}) \quad \text{for } k = 0, 1, 2, \dots$$

□

REMARK 8.5 A special case, in which $\rho(G'(x^*)) = 0$, is when $A(x) = F'(x)$. This corresponds to the iteration

$$x^{(k+1)} = x^{(k)} - (F'(x^{(k)}))^{-1}F(x^{(k)}), \quad (8.29)$$

which is Newton's method.

□

Theorem 8.5 leads to the following local convergence result for Newton's method.

THEOREM 8.6

Assume that $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ is Fréchet differentiable on an open neighborhood $S_0 \subset D$ of a point $x^* \in D$ for which $F(x^*) = 0$. Also, assume that $F'(x)$ is continuous at x^* and that $F'(x^*)$ is nonsingular. Then x^* is a point of attraction of Newton's method (8.29).

PROOF By Theorem 8.5, with $A(x) = F'(x)$ for $x \in S_0$, we conclude that $G(x) = x - (F'(x))^{-1}F(x)$ is well-defined on some ball $\bar{S}(x^*, \delta) \subset S_0$, $\delta > 0$. In addition, $\rho(G'(x^*)) = \sigma = 0$. Therefore, by Corollary 8.1, x^* is a point of attraction. □

8.3.2 Convergence Rate of Newton's Method

We now examine the rate of convergence of Newton iteration.

PROPOSITION 8.1

Assume that the hypotheses of Theorem 8.6 hold. Then, for the point of attraction of the Newton iteration (whose existence is guaranteed by Theo-

rem 8.6), we have

$$\lim_{k \rightarrow \infty} \frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} = 0. \quad (8.30)$$

Moreover, if for some constant $\hat{c}$,

$$\|F'(x) - F'(x^*)\| \leq \hat{c}\|x - x^*\| \quad (\text{that is, } F' \text{ is Lipschitz continuous}) \quad (8.31)$$

for all x in some neighborhood of x^* , then there exists a positive constant c such that

$$\|x^{(k+1)} - x^{(k)}\| \leq c\|x^{(k)} - x^*\|^2. \quad (8.32)$$

REMARK 8.6 If $\|x^{(k+1)} - x^*\|/\|x^{(k)} - x^*\| \leq \alpha$ for all k sufficiently large, the convergence is said to be linear. Equation (8.30) indicates Newton's method has superlinear convergence, and if (8.31) is satisfied, then Newton's method is quadratically convergent near x^* . $\square$

PROOF Recall that the fixed point iteration function is $G(x) = x - (F'(x))^{-1}F(x)$ for Newton's method. In Theorem 8.5, G was shown to be well-defined in some ball about x^* and the F-derivative of G was shown to exist at x^* . Then, for $x^{(k)}$ in the ball of attraction, $x^{(k+1)} = G(x^{(k)})$ implies

$$\lim_{k \rightarrow \infty} \frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} = \lim_{k \rightarrow \infty} \frac{\|G(x^{(k)}) - G(x^*) - G'(x^*)(x^{(k)} - x^*)\|}{\|x^{(k)} - x^*\|} = 0.$$

This follows from the fact $G'(x^*) = I - (F'(x^*))^{-1}F'(x^*) = 0$ and from the definition of Fréchet differentiability. Thus, (8.30) is valid.

Now, let k_0 be such that (8.31) holds in a ball containing $\{x^{(k)}\}_{k \geq k_0}$. For any such $k \geq k_0$, consider the convex set consisting of points between $x^{(k)}$ and x^* . Using (8.31) and Lemma 8.3 (given following this proof), we obtain:

$$\|F(x^{(k)}) - F(x^*) - F'(x^*)(x^{(k)} - x^*)\| \leq \frac{\hat{c}}{2}\|x^{(k)} - x^*\|^2. \quad (8.33)$$

Now,

$$\begin{aligned} \|x^{(k+1)} - x^*\| &= \|G(x^{(k)}) - x^*\| = \|x^{(k)} - (F'(x^{(k)}))^{-1}F(x^{(k)}) - x^*\| \\ &\leq \|(F'(x^{(k)}))^{-1}\{F(x^{(k)}) - F(x^*) - (F'(x^*))(x^{(k)} - x^*)\}\| \\ &\quad + \|(F'(x^{(k)}))^{-1}\{(F'(x^{(k)}) - F'(x^*))(x^{(k)} - x^*)\}\| \\ &\leq \|(F'(x^{(k)}))^{-1}\|(\frac{\hat{c}}{2} + \hat{c})\|x^{(k)} - x^*\|^2 \end{aligned}$$

by (8.31) and (8.33) and because $F(x^*) = 0$. Thus, with $A = F'$, the hypotheses of Theorem 8.5 are satisfied, so (8.25) (with k sufficiently large) implies that

$$\|(F'(x^{(k)}))^{-1}\| \leq 2\|F'(x^*)^{-1}\| = \text{a constant}.$$

This, in turn, implies (8.32). $\square$

LEMMA 8.3

Let $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ be continuously Fréchet differentiable on a convex set $D_0 \subset D$ and suppose for some constant $\alpha \geq 0$, F' satisfies

$$\|F'(u) - F'(v)\| \leq \alpha \|u - v\| \quad \text{for } u, v \in D_0. \quad (8.34)$$

Then, for any $x, y \in D_0$.

$$\|F(y) - F(x) - F'(x)(y - x)\| \leq \frac{\alpha}{2} \|x - y\|^2.$$

PROOF We showed in the proof of Theorem 8.3 (on page 444) that

$$F(y) - F(x) = \int_0^1 F'(x + s(y - x))(y - x) ds.$$

Thus,

$$\begin{aligned} \|F(y) - F(x) - F'(x)(y - x)\| &= \left\| \int_0^1 (F'(x + s(y - x)) - F'(x))(y - x) ds \right\| \\ &\leq \int_0^1 \|F'(x + s(y - x)) - F'(x)\| \|y - x\| ds \\ &\leq \alpha \int_0^1 s \|y - x\|^2 ds = \frac{\alpha}{2} \|y - x\|^2. \end{aligned}$$

$\square$

8.3.3 Example of Newton's Method

Here, we see an iterative method for computing the inverse of a nonsingular matrix. Let A be an $n \times n$ nonsingular matrix, and view A as an n^2 -vector (for example, by identifying the first row with the first n components, the second row with the second n components, etc.). Similarly, let x be an $n \times n$ matrix, use the notation x^{-1} to denote the inverse of x , and view x and x^{-1} as n^2 -vectors. With this notation, define

$$F(x) = x^{-1} - A.$$

(Then $F(A^{-1}) = 0$ so the solution to $F(x) = 0$ is $x^* = A^{-1}$).

What is $F'(x)$? To find $F'(x)$, we calculate

$$\lim_{\|y\| \rightarrow 0} \frac{\|F(x + y) - F(x) - F'(x)y\|}{\|y\|} = 0.$$

Proceeding with these computations, we obtain

$$\begin{aligned}
 & \lim_{\|y\| \rightarrow 0} \frac{\|F(x+y) - F(x) - F'(x)y\|}{\|y\|} \\
 &= \lim_{\|y\| \rightarrow 0} \frac{\|(x+y)^{-1} - A - x^{-1} + A - F'(x)y\|}{\|y\|} \\
 &= \lim_{\|y\| \rightarrow 0} \frac{\|(x+y)^{-1} - x^{-1} - F'(x)y\|}{\|y\|}. \tag{8.35}
 \end{aligned}$$

Now, observe that

$$\begin{aligned}
 (x+y) &= x(I + x^{-1}y), \quad \text{so} \\
 (x+y)^{-1} &= (I + x^{-1}y)^{-1}x^{-1}.
 \end{aligned}$$

Also, recall that

$$(I+B)^{-1} = I - B + B^2 - B^3 + \cdots \quad \text{if } \rho(B) < 1$$

(see Theorem 3.5 on page 102 and Proposition 3.8 on page 103), so

$$(I + x^{-1}y)^{-1} = I - x^{-1}y + (x^{-1}y)^2 - (x^{-1}y)^3 + \cdots \quad \text{if } \rho(x^{-1}y) < 1, \tag{8.36}$$

where $\rho(x^{-1}y) < 1$ whenever $\|y\|$ is small enough. Therefore, the limit in (8.35) is equal to

$$\begin{aligned}
 & \lim_{\|y\| \rightarrow 0} \frac{\|(I + x^{-1}y)^{-1}x^{-1} - x^{-1} - F'(x)y\|}{\|y\|} \\
 &= \lim_{\|y\| \rightarrow 0} \frac{\|(I - x^{-1}y + (x^{-1}y)^2 - \cdots)x^{-1} - x^{-1} - F'(x)y\|}{\|y\|} \\
 &= \lim_{\|y\| \rightarrow 0} \frac{\|(x^{-1} - x^{-1}yx^{-1} + \cdots) - x^{-1} - F'(x)y\|}{\|y\|} \\
 &= \lim_{\|y\| \rightarrow 0} \frac{\|-x^{-1}yx^{-1} + (x^{-1}y)^2x^{-1} \cdots - F'(x)y\|}{\|y\|} \\
 &= 0 \quad \text{provided } F'(x)y = -x^{-1}yx^{-1}.
 \end{aligned}$$

Thus, $F'(x)y = -x^{-1}yx^{-1}$. But we need $(F'(x))^{-1}$ for Newton's method. We have $y = -(F'(x))^{-1}x^{-1}yx^{-1}$, from which we obtain $(F'(x))^{-1}k = -xkx$, where $k = x^{-1}yx^{-1}$ and $y = +xkx$. Therefore, Newton's method has the form for this problem:

$$\begin{aligned}
 x^{(k+1)} &= x^{(k)} - (F'(x^{(k)}))^{-1}F(x^{(k)}) \\
 &= x^{(k)} + x^{(k)}F(x^{(k)})x^{(k)} \\
 &= x^{(k)} + x^{(k)}((x^{(k)})^{-1} - A)x^{(k)} = x^{(k)} + x^{(k)}(I - Ax^{(k)})
 \end{aligned}$$

or

$$x^{(k+1)} = 2x^{(k)} - x^{(k)}Ax^{(k)} \quad \text{for } k = 0, 1, 2, \dots, \tag{8.37}$$

and $x^{(k)} \rightarrow A^{-1}$ as $k \rightarrow \infty$. For this method, $x^{(k)}$ converges quadratically to A^{-1} . Notice that method (8.37) corresponds to method (3.82) described in section 3.4.11 (on page 167).

8.3.4 Local, Semilocal, and Global Convergence

In section 8.3.1, we proved a local convergence result for Newton's method (Theorem 8.6 on page 450). In the next two sections, we will consider two other well-known results for Newton's method. First, it is useful to understand that in a *local convergence result*, a solution x^* is assumed to exist, and it is shown that there is a neighborhood about x^* such that the iterates converge to x^* . In a *semilocal convergence result*, it is shown that, for a particular choice of initial values, the iterates converge to a solution x^* . Finally, in a *global convergence result*, it is shown that, for initial values on a large subset,² there is convergence to a solution x^* .

8.3.5 The Newton–Kantorovich Theorem

The following semilocal convergence result is perhaps the most widely-recognized convergence theorem associated with Newton's method.

THEOREM 8.7

(Newton–Kantorovich) Let $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ be F -differentiable on a convex set $D_0 \subset D$, let $\|\cdot\|$ be some norm, and assume that, for some constant $\gamma \geq 0$,

$$\|F'(x) - F'(y)\| \leq \gamma \|x - y\| \quad \text{for all } x, y \in D_0 \quad (8.38)$$

(That is, assume that F' is Lipschitz continuous on D_0 .) Suppose that $F'(x^{(0)})$ is invertible for any $x^{(0)} \in D_0$. Moreover, suppose that, for constants $\beta, \eta > 0$,

$$\|(F'(x^{(0)}))^{-1}\| \leq \beta \quad (8.39)$$

$$\|(F'(x^{(0)}))^{-1}F(x^{(0)})\| \leq \eta. \quad (8.40)$$

Also assume that

$$\alpha = \beta\gamma\eta \leq \frac{1}{2}. \quad (8.41)$$

Set

$$t^* = \frac{1 - (1 - 2\alpha)^{\frac{1}{2}}}{\beta\delta}, \quad (8.42)$$

and assume that

$$\overline{S}(x^{(0)}, t^*) = \left\{ x : \|x - x^{(0)}\| \leq t^* \right\} \subset D_0.$$

²or in the entire domain of F

Then, the Newton iterates

$$x^{(k+1)} = x^{(k)} - (F'(x^{(k)}))^{-1}F(x^{(k)}), \quad k = 0, 1, 2, \dots$$

are well-defined, remain in $\overline{S}(x^{(0)}, t^*)$, and converge to a solution x^* of $F(x) = 0$ which is unique in $\overline{S}(x^{(0)}, t^*)$.

Moreover, we have the error estimate

$$\|x^{(k)} - x^*\| \leq \frac{(2\alpha)^{2^k}}{\beta\gamma 2^k} \quad k = 0, 1, 2, \dots \quad (8.43)$$

PROOF See [70].

□

8.3.6 A Global Convergence Result for Newton's Method

Global convergence of Newton's method (existence of a unique solution to which Newton's method converges from any starting point) occurs only under special circumstances. However, these circumstances occur commonly enough in practice to justify studying when such global convergence occurs.

Various possible sets of assumptions lead to global convergence of Newton's method. We now present an example of one possible global convergence theorem. We first need a definition and a lemma.

In the remainder of this section, inequalities between vectors are interpreted componentwise; for example, $v \geq w$ means $v_i \geq w_i$, $1 \leq i \leq n$. Similarly, matrix inequalities are interpreted componentwise; for example, if A is an n by n matrix, then $A \geq 0$ means $a_{ij} \geq 0$ for $1 \leq i \leq n$ and $1 \leq j \leq n$.

DEFINITION 8.7 F is convex on a convex set D_0 if

$$\lambda F(x) + (1 - \lambda)F(y) \geq F(\lambda x + (1 - \lambda)y)$$

for all $\lambda \in [0, 1]$ and $x, y \in D_0$.

We can now present

LEMMA 8.4

Let $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ be F -differentiable on the convex set $D_0 \subset D$. Then F is convex on D_0 if and only if

$$F(y) - F(x) \geq F'(x)(y - x) \quad \text{for } x, y \in D_0. \quad (8.44)$$

PROOF First suppose that (8.44) holds. Fix $x, y \in D_0$ and $\lambda \in [0, 1]$, and set $z = \lambda x + (1 - \lambda)y$. Since D_0 is convex, $z \in D_0$ and (8.44) imply

$$(i) \quad F(x) - F(z) \geq F'(z)(x - z) \quad \text{and}$$

$$(ii) \quad F(y) - F(z) \geq F'(z)(y - z).$$

Multiplying (i) by λ and (ii) by $(1 - \lambda)$ and adding, we obtain

$$\begin{aligned} \lambda F(x) + (1 - \lambda)F(y) - F(z) &\geq \lambda F'(z)(x - z) + (1 - \lambda)F'(z)(y - z) \\ &= F'(z)[\lambda x + (1 - \lambda)y - z] = 0. \end{aligned}$$

Hence,

$$\lambda F(x) + (1 - \lambda)F(y) \geq F(z) = F(\lambda x + (1 - \lambda)y),$$

which shows that F is convex.

Now suppose that F is convex on D_0 , and let $0 \leq \lambda \leq 1$. Then we can write

$$F(\lambda y + (1 - \lambda)x) \leq \lambda F(y) + (1 - \lambda)F(x)$$

in the form

$$\frac{1}{\lambda}(F(x + \lambda(y - x)) - F(x)) \leq F(y) - F(x). \quad (8.45)$$

By F -differentiability of F , it follows that the left side tends to $F'(x)(y - x)$ as $\lambda \rightarrow 0$. Also, since, if $a(\lambda), b \in \mathbb{R}^n$ are such that $a(\lambda) \leq b$ for all $\lambda \neq 0$ and $\lim_{\lambda \rightarrow 0} a(\lambda) = a$, it follows that $a \leq b$ (componentwise). Thus, (8.45) implies (8.44). $\square$

We can now state and prove an example of a global convergence result for Newton's method.

THEOREM 8.8

Let $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$ be continuously Fréchet-differentiable and convex over all of $\mathbb{R}^n$. In addition, suppose that $(F'(x))^{-1}$ exists for all $x \in \mathbb{R}^n$ and $(F'(x))^{-1} \geq 0$ for all $x \in \mathbb{R}^n$. Let $F(x) = 0$ have a solution x^* . Then x^* is unique, and the Newton iterates $x^{(k+1)} = x^{(k)} - (F'(x^{(k)}))^{-1}F(x^{(k)})$ converge to x^* for any initial choice $x^{(0)} \in \mathbb{R}^n$. Moreover, for all $k > 0$,

$$x^* \leq x^{(k+1)} \leq x^{(k)} \quad \text{for } k = 1, 2, \dots \quad (8.46)$$

(Throughout this theorem statement, the inequalities are interpreted componentwise.)

PROOF First, we show by induction that (8.46) holds. Let $x^{(0)} \in \mathbb{R}^n$ be arbitrary. By hypothesis, all the Newton iterates are well-defined, and

$$x^{(1)} = x^{(0)} - (F'(x^{(0)}))^{-1}F(x^{(0)}).$$

Then, by Lemma 8.4,

$$F(x^{(1)}) - F(x^{(0)}) \geq F'(x^{(0)})(x^{(1)} - x^{(0)}) = -F(x^{(0)}),$$

so $F(x^{(1)}) \geq 0$. But again by Lemma 8.4,

$$0 = F(x^*) \geq F(x^{(1)}) + F'(x^{(1)})(x^* - x^{(1)}).$$

It then follows from $(F'(x^{(1)}))^{-1} \geq 0$ that

$$0 \geq (F'(x^{(1)}))^{-1} F(x^{(1)}) + (x^* - x^{(1)}).$$

Thus, we conclude that $x^* \leq x^{(1)}$. For general k , it follows exactly as above, and we obtain

$$x^* \leq x^{(k)} \quad \text{and} \quad F(x^{(k)}) \geq 0 \quad \text{for } k = 1, 2, \dots.$$

But

$$x^{(k+1)} = x^{(k)} - (F'(x^{(k)}))^{-1} F(x^{(k)}) \leq x^{(k)}.$$

Therefore, 8.46 holds.

Now, we need to show that $x^{(k)} \rightarrow x^*$ as $k \rightarrow \infty$. First, note that $x_i^{(k)}$ is monotonically decreasing and bounded below by x_i^* for each i , $1 \leq i \leq n$. Thus, $x^{(k)} \rightarrow y = (y_1, y_2, \dots, y_n)^T$ as $k \rightarrow \infty$. Furthermore, since $F'(x)$ is a continuous function of x ,

$$\begin{aligned} F(y) &= \lim_{k \rightarrow \infty} F(x^{(k)}) = \lim_{k \rightarrow \infty} F'(x^{(k)})(x^{(k+1)} - x^{(k)}) \quad (\text{Newton iterates}) \\ &= F'(y)0 = 0. \end{aligned}$$

Hence, y is a solution of $F(x) = 0$. But $F(x) = 0$ has only one solution x^* . To see this, let x^* and y^* be two solutions; then, by Lemma 8.4,

$$0 = F(x^*) - F(y^*) \geq F'(x^*)(y^* - x^*),$$

and multiplying both sides by $(F'(x^*))^{-1} \geq 0$ gives $y^* \leq x^*$. Reversing the roles of x^* and y^* , we obtain $x^* \leq y^*$ and thus $x^* = y^*$. $\square$

Example 8.3

(One Dimension) Let $F(x) = x + e^x + a$ for some $a \in \mathbb{R}$. Then F is convex for all $x \in \mathbb{R}$. To see this, one can see, e.g. from Taylor's Theorem, that $e^z \geq 1 + z$ for all $z \in \mathbb{R}$. Thus, $e^{(y-x)} - 1 \geq y - x$, so $e^y - e^x \geq e^x(y - x)$. Therefore, $y + e^x + a - x - e^x - a \geq (1 + e^x)(y - x)$, and we have $F(y) - F(x) \geq F'(x)(y - x)$ for $x, y \in \mathbb{R}$. Now, consider $F'(x) = 1 + e^x$. We have

$$(F'(x))^{-1} = \frac{1}{1 + e^x} \geq 0 \quad \text{for all } x \in \mathbb{R}.$$

Finally, the Intermediate Value Theorem implies that there is an $x^* \in \mathbb{R}$ such that $F(x^*) = 0$. Hence, Theorem 8.8 can be applied, namely, the iterates defined by

$$x^{(k+1)} = x^{(k)} - \left(\frac{1}{1 + e^{x^{(k)}}} \right) (x^{(k)} + e^{x^{(k)}} + a)$$

converge to x^* . $\square$

8.3.7 Practical Considerations

The following algorithm combines Newton's method with a few practical considerations.

ALGORITHM 8.1

(Newton's method)

INPUT:

- (a) an initial guess $x^{(0)}$;
- (b) a maximum number of iterations M .
- (c) a domain stopping tolerance ϵ_d and a range stopping tolerance ϵ_r

OUTPUT: either "success" or "failure." If "success," then also output the number of iterations k and the approximation $x^{(k+1)}$ to the solution x^* .

1. "success" $\leftarrow$ "false".

2. FOR $k = 0$ to M .

(a) Evaluate $F'(x^{(k)})$. (That is, evaluate the corresponding n^2 partial derivatives at $x^{(k)}$.)

(b) Solve $F'(x^{(k)})v^{(k)} = -F(x^{(k)})$ for $v^{(k)}$.

 ○ IF $F'(x^{(k)})v^{(k)} = -F(x^{(k)})$ cannot be solved (such as when $F'(x^{(k)})$ is numerically singular) THEN EXIT.

(c) $x^{(k+1)} \leftarrow x^{(k)} + v^{(k)}$.

(d) IF ($\|v^{(k)}\| < \epsilon_d$ or $\|F(x^{(k+1)})\| < \epsilon_r$) THEN

 i. "success" $\leftarrow$ "true".

 ii. EXIT.

END FOR

END ALGORITHM 8.1.

8.3.7.1 Advantages of Newton's Method

- (a) If $F'(x^*)$ is nonsingular, then a domain of attraction exists. (Thus, if a Newton iterate lands in the attraction ball, the successive iterates will remain in the ball and eventually converge to x^* .)
- (b) The convergence is generally superlinear, and if F' satisfies a Lipschitz condition at x^* , the convergence is quadratic. (Recall that, in quadratic convergence, the number of significant digits in $x^{(k)}$ as an approximation to x^* is doubled each iteration.)

- (c) Newton's method is "self-correcting" in the sense that $x^{(k+1)}$ only depends on F and $x^{(k)}$, so bad effects (such as the effects of roundoff error) from previous iterations are not propagated.³

8.3.7.2 Disadvantages of Newton's Method

- (a) The attraction ball may be very small, so a good initial approximation to x^* may be required.
- (b) We need to solve a linear system of size n at each step, which requires n^3 work for a dense system.
- (c) The Jacobian matrix is required at each step, which requires evaluation of n^2 scalar functions $\frac{\partial f_i}{\partial x_j}$ for a dense system.⁴

8.3.7.3 Modifications to Newton's Method

How can we make the procedure faster or more computationally convenient? Some possibilities are:

1. Approximate the partial derivatives in $F'(x^{(k)})$, e.g.

$$\begin{aligned} & \frac{\partial f_i(x^{(k)})}{\partial x_j} \\ & \simeq \frac{f_i(x_1^{(k)}, x_2^{(k)}, \dots, x_j^{(k)} + h, x_{j+1}^{(k)}, \dots, x_n^{(k)}) - f_i(x_1^{(k)}, \dots, x_n^{(k)})}{h}, \end{aligned}$$

where h is small. However, it can be proved, under certain conditions on F , that the method is only linearly convergent.⁵ Nonetheless, this method may be reasonable for *black box systems*, that is, systems for which only the values of f can be obtained (and not the equations, or the computer program that evaluates f).

2. Use automatic (algorithmic differentiation) (as we introduced in Section 6.2, starting on page 327) to compute F' .
3. Solve $F'(x^{(k)})y^{(k)} = F(x^{(k)})$ approximately.

As associated software improves, algorithmic differentiation (item 2) is increasingly replacing finite-difference approximations (item 1) in Newton's

³This is in contrast to, say, an unstable method for solving an initial value problem.

⁴However, these partial derivatives do not necessarily need to be programmed by hand; moreover, automatic differentiation techniques can take advantage of the structure of the system to reduce the total amount of computation required.

⁵Finite differences have been used in the past to remove the need to manually compute and program partial derivatives. However, this reason for using finite differences has disappeared for many applications, for which automatic differentiation or derivatives produced with computer algebra systems can now be used.

method. Not only is quadratic convergence improved, but considerably less than n^3 operations are required to compute the Jacobian matrix when the system is structured (i.e., when the Jacobian matrix has a sparse structure).

Approximately solving the system (item 3), especially with iterative methods, remains a popular technique, especially for very large, structured systems of equations. In the remainder of this section, we consider this in more detail. First, consider iterative solution of the system $Ax = b$. Recall that the SOR method can be written in the form:

$$x^{(m+1)} = x^{(m)} - \sigma(D - \sigma L)^{-1}(Ax^{(m)} - b) \quad \text{for } m = 0, 1, 2, \dots \quad (8.47)$$

with $x^{(0)} = x_0$ given, where

$$A = D - L - U. \quad (8.48)$$

Recall that when $\sigma = 1$, we obtain the Gauss–Seidel iterative method.

One way to use SOR for nonlinear problems is by approximately solving the linear system present at each step of Newton's method. In this case, the primary iteration is Newton's method and the secondary is SOR. We call this the Newton-SOR method, which we now describe.

In Newton's method, $x^{(k+1)} = x^{(k)} - (F'(x^{(k)}))^{-1}F(x^{(k)})$ for $k = 0, 1, 2, \dots$. This can be written as

$$F'(x^{(k)})x^{(k+1)} = F'(x^{(k)})x^{(k)} - F(x^{(k)}). \quad (8.49)$$

We solve (8.49) approximately using SOR. To do this we decompose the Jacobian matrix $F'(x^{(k)})$ as

$$F'(x^{(k)}) = D_k - L_k - U_k, \quad (8.50)$$

and specify some relaxation parameter σ , $0 < \sigma_k < 2$. To apply SOR to (8.49), we denote the m -th SOR iterate by $x^{k,m}$, $m = 0, 1, 2, \dots$ and apply (8.47) with $A = F'(x^{(k)})$, $b_k = F'(x^{(k)})x^{(k)} - F(x^{(k)})$, to obtain

$$x^{k,m} = x^{k,m-1} - \sigma_k(D_k - \sigma_k L_k)^{-1}(F'(x^{(k)})x^{k,m-1} - b_k) \quad (8.51)$$

for $m = 1, 2, 3, \dots$. A natural choice for $x^{k,0}$ is to let it equal x^k of the previous Newton iterate. Finally, we assume the SOR iterations are terminated after m_k iterations, and set $x^{k+1} = x^{k,m_k}$.

How is m_k chosen? We could choose m_k by terminating the SOR iterations by a convergence criterion such as $\|x^{k,m} - x^{k,m-1}\| \leq \epsilon$ for some specified ϵ ; then, m_k varies with k . We could also specify m_k in advance. The simplest choice is $m_k = 1$, which leads to the *1-step Newton–SOR iteration*:

$$x^{(k+1)} = x^{(k)} - \sigma_k(D_k - \sigma_k L_k)^{-1}F(x^k), \quad k = 0, 1, \dots \quad (8.52)$$

Furthermore, if $\sigma_k = 1$, method (8.51) is called the *Newton–Gauss–Seidel Method*. Note that the 1-step Newton–Gauss–Seidel method only requires

evaluation of $D_k - \sigma_k L_k$, i.e., the evaluation of $\frac{n(n+1)}{2}$ partial derivatives at $x^{(k)}$ while the m -step Newton–SOR method ($m > 1$) requires n^2 partial derivatives at $x^{(k)}$, assuming the system is dense.

There is a second way of extending an iterative method for linear systems to nonlinear systems. Consider, first, the Jacobi method, perhaps the simplest iterative scheme for linear systems. Recall for $Ax = b$, the Jacobi method consists of

$$x_i^{(k+1)} = -\frac{1}{a_{ii}} \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij} x_j^{(k)} + \frac{b_i}{a_{ii}} \quad i = 1, 2, \dots, n \quad \text{for } k = 0, 1, 2, \dots, \quad (8.53)$$

with $x_i^{(0)}$ given.

Equation (8.53) can be interpreted as solving approximately the i -th equation of the system $Ax = b$ for unknown x_i , holding fixed all other unknowns x_j , $j \neq i$, at the k -th level, i.e., $x_j^{(k)}$.

Consider now the nonlinear map $F: D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$, where we seek x^* such that $F(x^*) = 0$ and $F(x) = (f_1(x), \dots, f_n(x))^T$. The analog of the linear Jacobi method is the *nonlinear Jacobi method*, where the unknowns x_j , $j \neq i$, at the level $x_j^{(k)}$ are kept fixed, the i -th equation for $F(x) = 0$ is solved for x_i . That is, we solve

$$f_i(x_1^{(k)}, x_2^{(k)}, \dots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k)}, \dots, x_n^{(k)}) = 0 \quad (8.54)$$

for x_i for $1 \leq i \leq n$ (substituting the current values $x_j^{(k)}$, $j \neq i$). We call the resulting vector $x = (x_1, x_2, \dots, x_n)^T$, thus defining the next iterate:

$$x^{(k+1)} = (x_1^{(k+1)}, x_2^{(k+1)}, \dots, x_n^{(k+1)})^T = (x_1, x_2, \dots, x_n)^T.$$

For each i , (8.54) is just a single nonlinear equation with one unknown x_i , and can be solved, for example, by applying m_k steps of Newton's method (applied to the scalar nonlinear equation). Thus, the j -th Newton step has the form:

$$x_i^{(j)} = x_i^{(j-1)} - \frac{f_i(x_1^{(k)}, x_2^{(k)}, \dots, x_{i-1}^{(k)}, x_i^{(j-1)}, x_{i+1}^{(k)}, \dots, x_n^{(k)})}{\frac{\partial f_i(x_1^{(k)}, \dots, x_i^{(j-1)}, \dots, x_n^{(k)})}{\partial x_i}}, \quad (8.55)$$

where $j = 1, 2, \dots, m_k$, and $x_i^{(0)}$ is taken as $x_i^{(k)}$. We are thus led to the Jacobi–Newton Method.

For example, the 1-step *Jacobi–Newton Method* involves applying for each i , $1 \leq i \leq n$, one step of Newton's method for approximately solving (8.54) for each x_i .

In an analogous manner, we can define Gauss–Seidel–Newton or SOR–Newton methods. For example, in *Gauss–Seidel–Newton methods*, for the

i -th equation, we solve

$$f_i(\underbrace{x_1^{(k+1)}, x_2^{(k+1)}, \dots, x_i^{(k+1)}}_{\text{Note: } k+1}, x_i, \underbrace{x_{i+1}^{(k)}, \dots, x_n^{(k)}}_{\text{Note: } k}) = 0 \quad (8.56)$$

for x_i by applying m_k steps of Newton's method, and we call the result $x_i^{(k+1)}$. More generally, if after finding x_i from (8.56), we set

$$x_i^{(k+1)} = x_i^{(k)} + \sigma_k(x_i - x_i^{(k)}) \quad (8.57)$$

for some parameter σ_k , we obtain the SOR–Newton Method. The one-step SOR method has the form

$$x_i^{(k+1)} = x_i^{(k)} - \sigma_k f_i(x^{(k,i)}) \left(\frac{\partial f_i(x^{(k,i)})}{\partial x_i} \right)^{-1} \quad (8.58)$$

where $x^{(k,i)} = (x_1^{(k+1)}, x_2^{(k+1)}, \dots, x_{i-1}^{(k+1)}, x_i^{(k)}, \dots, x_n^{(k)})^T$. (See Exercise 11 below.)

Hence, the 1-step SOR–Newton method requires, at each step k , the evaluation of the n functions $f_i(x^{(k,i)})$ as well as the n derivatives $\partial f_i(x^{(k,i)})/\partial x_i$. (Contrast this with the number of component function evaluations required for Newton–SOR method.)

REMARK 8.7 Note the difference between e.g. the *Newton–Gauss–Seidel Method*, in which we solve the linear system arising from the multivariate Newton method with Gauss–Seidel iteration, and the *Gauss–Seidel–Newton Method*, in which, in principle we solve the i -th nonlinear equation for the i -th variable without first replacing it by a linear approximation. $\square$

REMARK 8.8 For general matrices, iterative methods for solving linear systems of equations need to be preconditioned first. However, certain applications occurring in practice do not require preconditioning. For example, the linear systems arising from discretization of the heat equation do not require preconditioning for the Gauss–Seidel (and Jacobi and SOR) methods to converge. If “mild” nonlinearities (that is, linearities that are not large in relation to the other terms in the equations) are introduced into such systems, we obtain nonlinear systems for which the Gauss–Seidel–Newton method will converge. Convergence, even local convergence, cannot be expected for general nonlinear systems when the Jacobi–Newton method, Gauss–Seidel–Newton method, or SOR–Newton method is used. $\square$

REMARK 8.9 These composite methods, e.g. SOR–Newton, do not possess the superlinear convergence rate of Newton's method. In fact, they converge linearly, if they converge. However, other convergence results (e.g.

local, global, etc.) for these composite methods that are similar to Newton's method can be obtained under certain assumptions. $\square$

8.4 Multivariate Interval Newton Methods

Multivariate interval Newton methods are similar to univariate interval Newton methods (as presented in Section 2.5, starting on page 54), in the sense that they provide rigorous bounds on solutions, in addition to existence and uniqueness proofs [44, 62]. Because of this, multivariate interval Newton methods have a good potential for computing mathematically rigorous bounds on a solution to a nonlinear system of equations, given an approximate solution (computed, say, by a point Newton method). Interval Newton methods are also used as parts of more involved algorithms to find all solutions to a nonlinear system, or for global optimization. (See the section on branch and bound algorithms, on page 523 below.)

Most multivariate interval Newton methods follow a form similar to that of the multivariate point method seen in Formula 8.22. To explain this, we introduce two preliminary definitions.

DEFINITION 8.8 Suppose $F : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$, and suppose $\mathbf{x} \in D$ is an interval n vector (i.e., a “box”). Then an interval matrix $\mathbf{A}$ is said to be a Lipschitz matrix for F over $\mathbf{x}$, if and only if, for each $x \in \mathbf{x}$ and $y \in \mathbf{x}$, there is an $A \in \mathbf{A}$ such that

$$F(y) - F(x) = A(y - x).$$

You will show in Exercise 16 (on page 484 below) that matrices formed from interval extensions of the partial derivatives are Lipschitz matrices.

Example 8.4

Suppose $F(x) = (f_1(x_1, x_2), f_2(x_1, x_2))^T$, where

$$\begin{aligned} f_1(x_1, x_2) &= x_1^2 - x_2^2 - 1, \\ f_2(x_1, x_2) &= 2x_1x_2. \end{aligned}$$

Then the Jacobian matrix is

$$F'(x) = \begin{pmatrix} 2x_1 & -2x_2 \\ 2x_2 & 2x_1 \end{pmatrix},$$

and a Lipschitz matrix for F over the box $\mathbf{x} = ([-0.1, 0.1], [0.9, 1.1])^T$ is

$$\mathbf{A} = \mathbf{F}'(\mathbf{x}) = \begin{pmatrix} 2[-0.1, 0.1] & -2[0.9, 1.1] \\ 2[0.9, 1.1] & 2[-0.1, 0.1] \end{pmatrix} = \begin{pmatrix} [-0.2, 0.2] & [-2.2, -1.8] \\ [1.8, 2.2] & [-0.2, 0.2] \end{pmatrix}.$$

(We use $\mathbf{F}'(\mathbf{x})$ to denote an elementwise interval evaluation of the Jacobian matrix for F .) $\square$

DEFINITION 8.9 Suppose $F : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$, suppose $\mathbf{x} \in D$ is an interval n -vector, and suppose $\tilde{x} \in D$. Then an interval matrix $\mathbf{A}$ is said to be a slope matrix for F at $\tilde{x}$ over $\mathbf{x}$ if and only if for each $x \in \mathbf{x}$, there is an $A \in \mathbf{A}$ such that

$$F(x) - F(\tilde{x}) = A(x - \tilde{x}).$$

Slope matrices can have narrower entries than Lipschitz matrices, so they may lead to results when Lipschitz matrices do not. However, slope matrices can be somewhat more complicated to compute than Lipschitz matrices, and they are trickier to use in processes that prove uniqueness.

The general interval Newton method can now be stated as

DEFINITION 8.10 Suppose $F : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$, suppose $\mathbf{x} \in D$ is an interval n -vector, and suppose that $\mathbf{A}$ is an interval matrix such that either

1. $\mathbf{A}$ is a slope matrix for F at $\tilde{x}$ over $\mathbf{x}$, or
2. $\tilde{x} \in D$, and $\mathbf{A}$ is a Lipschitz matrix for F over $\mathbf{x}$.

Then a multivariate interval Newton operator F is any mapping $\mathbf{N}(F, \mathbf{x}, \tilde{x})$ from the set of ordered pairs $(\mathbf{x}, \tilde{x})$ of interval n -vectors $\mathbf{x}$ and point n -vectors $\tilde{x}$ to the set of interval n -vectors, such that

$$\tilde{x} \leftarrow \mathbf{N}(F, \mathbf{x}, \tilde{x}) = \tilde{x} + \mathbf{v}, \quad (8.59)$$

where $\mathbf{v} \in \mathbb{R}^n$ is any box that bounds the solution set to the linear interval system

$$\mathbf{A}\mathbf{v} = -F(\tilde{x}). \quad (8.60)$$

REMARK 8.10 In implementations of interval Newton methods on computers, the vector $F(\tilde{x})$ is evaluated using interval arithmetic, even though the value sought is at a point. This is to take account of roundoff error, so the results will be mathematically rigorous. $\square$

An immediate consequence of Definition 8.10 is

PROPOSITION 8.2

Suppose F has continuous first-order partial derivatives, and $\mathbf{N}(F, \mathbf{x}, \tilde{x})$ is the image under an interval Newton method of the box $\mathbf{x}$. Then any solutions $x^* \in \mathbf{x}$ of $F(x) = 0$ must also lie in $\mathbf{N}(F, \mathbf{x}, \tilde{x})$.

PROOF The proof is a consequence of the definition of a Lipschitz matrix or of a slope matrix, and is left as Exercise 17 on page 485 below. $\square$

A uniqueness theorem can be stated in general for any interval Newton operator. We have

THEOREM 8.9

Suppose F and $N(F, \mathbf{x}, \tilde{x})$ are as in Definition 8.10, suppose that $N(F, \mathbf{x}, \tilde{x}) \subseteq \mathbf{x}$, suppose that $\mathbf{A}$ is chosen to be a Lipschitz matrix for F over $\mathbf{x}$, and suppose that there exists an $x \in \mathbf{x}$ such that $F(x) = 0$. Then there is no other $y \in \mathbf{x}$ with $F(y) = 0$ (that is, $\mathbf{x}$ is unique).

PROOF $N(F, \mathbf{x}, \tilde{x}) \subseteq \mathbf{x}$ implies that the interval enclosure $\mathbf{v}$ to the solution set to (8.60) is bounded. That, in turn implies that every $A \in \mathbf{A}$ is nonsingular. That said, assume that there is a $y \in \mathbf{x}$, $y \neq x$ with $F(y) = 0$. Then

$$0 = F(y) - F(x) = A(y - x) \quad \text{for some } A \in \mathbf{A},$$

since $\mathbf{A}$ is a Lipschitz set for F over $\mathbf{x}$. However, this contradicts the fact that A must be nonsingular. Therefore, $\mathbf{x}$ must be unique. $\square$

We now study some specific techniques for bounding the solution set of (8.60). In the process of studying practical aspects, we will relate multivariate interval Newton methods to the Gauss–Seidel method, to the nonlinear Gauss–Seidel method, to the contraction mapping theorem, to the Brouwer fixed point theorem, and to the Kantorovich theorem.

8.4.1 The Nonlinear Interval Gauss–Seidel Method

The nonlinear interval Gauss–Seidel method can be viewed in two ways: we can view it either as an interval version of the Newton–Gauss–Seidel method (where we are solving a linear system of equations in n variables) or as an interval version of the Gauss–Seidel–Newton method (where we are solving n nonlinear systems of one variable). Each of these views is advantageous for revealing different properties of the method.

8.4.1.1 As Newton–Gauss–Seidel

If we view the nonlinear interval Gauss–Seidel method as an interval version of the Newton–Gauss–Seidel method, then we will use the interval Gauss–Seidel method of Section 3.4.5 (starting on page 153) to bound the solution set to (8.60). In particular, if we apply the iteration scheme (3.58) (the interval version of the Gauss–Seidel method, on page 154) to the interval linear system

(8.60), assuming we precondition (8.60) to

$$(Y\mathbf{A})v = -YF(\tilde{x}),$$

we obtain

$$\begin{aligned} \mathbf{v}^{(0)} &\leftarrow \mathbf{x} - \tilde{x}, \\ \mathbf{v}_i^{(k+1)} &\leftarrow \frac{1}{(Y\mathbf{A})_{ii}} \left\{ -(YF(\tilde{x}))_i - \sum_{j=1}^{i-1} (Y\mathbf{A})_{ij} \mathbf{v}_j^{(k+1)} - \sum_{j=i+1}^n (Y\mathbf{A})_{ij} \mathbf{v}_j^{(k)} \right\} \\ &\text{for } i = 1, 2, \dots, n. \end{aligned} \tag{8.61}$$

For example, we can use an interval derivative matrix as a Lipschitz matrix. In that case, let $\partial \mathbf{f}_i / \partial \mathbf{x}_j(\mathbf{x})$ denote an interval enclosure of the range of the j -th partial derivative $\partial f_i / \partial x_j$ of f_i over $\mathbf{x}$ (such as can be obtained by evaluating an expression for $\partial f_i / \partial x_j$ with interval arithmetic), and denote by $\mathbf{F}'(\mathbf{x})$ the corresponding matrix. Then,

$$\mathbf{A} = (\mathbf{a}_{ij}) = \left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_j}(\mathbf{x}) \right) = \mathbf{F}'(\mathbf{x}).$$

If we further replace $\mathbf{v}_j^{(k)}$ by $\mathbf{x}_j - \tilde{x}_j$ in the iteration equation (8.61), then solve for $\mathbf{x}_i^{(k+1)}$, we obtain

$$\begin{aligned} \mathbf{x}_i^{(k+1)} &\leftarrow \tilde{x}_i^{(k)} - \frac{1}{(Y\mathbf{F}'(\mathbf{x}^{(k)}))_{ii}} \\ &\quad \cdot \left\{ (YF(\tilde{x}))_i + \sum_{j=1}^{i-1} (Y\mathbf{F}'(\mathbf{x}^{(k)}))_{ij} \left(\mathbf{x}_j^{(k+1)} - \tilde{x}_j^{(k)} \right) \right. \\ &\quad \left. + \sum_{j=i+1}^n (Y\mathbf{F}'(\mathbf{x}^{(k)}))_{ij} \left(\mathbf{x}_j^{(k)} - \tilde{x}_j^{(k)} \right) \right\} \\ &\text{for } i = 1, 2, \dots, n. \end{aligned} \tag{8.62}$$

In (8.62), $\tilde{x}^{(k+1)}$ can be chosen to be the midpoint of $\mathbf{x}^{(k+1)}$, although other choices are possible, and sometimes advisable.

8.4.1.2 As a Univariate Method with Uncertainty

The iteration (8.62) can also be derived with a multivariate version of the mean value extension $\mathbf{f}_{\text{mv}}$, considered in Problem 26 on page 33. We have

PROPOSITION 8.3

Suppose $f_i : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$, $\mathbf{x} \subseteq D$ is an interval n -vector, f_i has continuous partial derivatives, $\partial \mathbf{f}_i / \partial \mathbf{x}_j(\mathbf{x})$ denotes an interval enclosure for the range

of $\partial f_i / \partial x_j$ over $\mathbf{x}$, for $1 \leq j \leq n$, and $\tilde{x} \in \mathbf{x}$; further, define

$$\mathbf{f}_{i,\text{mv}}(\mathbf{x}) = f_i(\tilde{x}) + \sum_{j=1}^n \frac{\partial f_i}{\partial x_j}(\mathbf{x})(x_j - \tilde{x}_j). \quad (8.63)$$

Then $\mathbf{f}_{i,\text{mv}}(\mathbf{x})$ contains the range of f_i over $\mathbf{x}$.

PROOF The proof follows directly from the multivariate mean value theorem (Theorem 8.1 on page 441); you will fill in the details in Exercise 20 on page 485. $\square$

REMARK 8.11 $\mathbf{f}_{i,\text{mv}}$ is called a *multivariate mean value interval extension* of f_i . $\square$

REMARK 8.12 Slope matrices, as in Definition 8.9, can be used instead of interval enclosures to partial derivatives. In that case, $\tilde{x}$ need not necessarily be in $\mathbf{x}$. $\square$

With the multivariate mean value extension, we can consider f_i (or, for preconditioned systems, $(YF)_i$) to be a function of x_i , with uncertainty in its values introduced by the variables x_j , $j \neq i$. Base the multivariate mean value expansion about the point

$$\tilde{\mathbf{x}} = (\tilde{x}_1, \dots, \tilde{x}_{i-1}, t, \tilde{x}_{i+1}, \dots, \tilde{x}_n),$$

so that the mean value extension becomes

$$\begin{aligned} \varphi_i(t) &= (YF)_i(x_1, \dots, x_{i-1}, t, x_{i+1}, \dots, x_n) \\ &\in (YF)_i(\tilde{\mathbf{x}}) + \frac{\partial (YF)_i}{\partial x_i}(\mathbf{x})(t - t) + \sum_{\substack{j=1 \\ j \neq i}}^n \frac{\partial (YF)_i}{\partial x_j}(\mathbf{x})(x_j - \tilde{x}_j) \\ &= (YF)_i(\tilde{x}_1, \dots, \tilde{x}_{i-1}, t, \tilde{x}_{i+1}, \dots, \tilde{x}_n) + \mathbf{I}, \end{aligned} \quad (8.64)$$

where $\tilde{\mathbf{x}} = (\tilde{x}_1, \dots, \tilde{x}_{i-1}, t, \tilde{x}_{i+1}, \dots, \tilde{x}_n)^T$ and where we interpret

$$\begin{aligned} \mathbf{I} &= \frac{\partial (YF)_i}{\partial x_i}(\mathbf{x})(t - t) + \sum_{\substack{j=1 \\ j \neq i}}^n \frac{\partial (YF)_i}{\partial x_j}(\mathbf{x})(x_j - \tilde{x}_j) \\ &= \sum_{\substack{j=1 \\ j \neq i}}^n \frac{\partial (YF)_i}{\partial x_j}(\mathbf{x})(x_j - \tilde{x}_j) \end{aligned}$$

as an interval of uncertainty that is “constant” with respect to the variable t . Then, identifying φ_i in (8.64) with f and identifying t with x in the derivation of the univariate interval Newton method (Equations (2.10) and (2.11) on page 54), we obtain the interval Gauss–Seidel method as in Equation (8.62).

8.4.1.3 Existence and Uniqueness Verification Theory

Existence and uniqueness theory for the interval Gauss–Seidel method and other interval Newton methods is covered in detail in [62]. We presented⁶ a relatively simple proof of the existence-proving properties of the interval Gauss–Seidel method in [44]. We present this proof here, since it is based on Miranda’s theorem, useful on its own.

THEOREM 8.10

(Miranda’s Theorem) Suppose

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{pmatrix} = \begin{pmatrix} [\underline{x}_1, \bar{x}_1] \\ [\underline{x}_2, \bar{x}_2] \\ \vdots \\ [\underline{x}_n, \bar{x}_n] \end{pmatrix} \in \mathbb{IR}^n$$

is an interval n -vector, and define the faces of $\mathbf{x}$ by

$$\begin{aligned} \mathbf{x}_{\underline{i}} &= (\mathbf{x}_1, \dots, \mathbf{x}_{i-1}, \underline{x}_i, \mathbf{x}_{i+1}, \dots, \mathbf{x}_n)^T \quad \text{and} \\ \mathbf{x}_{\bar{i}} &= (\mathbf{x}_1, \dots, \mathbf{x}_{i-1}, \bar{x}_i, \mathbf{x}_{i+1}, \dots, \mathbf{x}_n)^T. \end{aligned}$$

Further, let $f = (f_1, \dots, f_n)^T : \mathbf{x} \rightarrow \mathbb{R}^n$ be continuous, and denote by $\mathbf{f}_i^u(\mathbf{y})$ the range of f_i over an interval vector $\mathbf{y} \in \mathbb{IR}^n$. If

$$\mathbf{f}_i^u(\mathbf{x}_{\underline{i}})\mathbf{f}_i^u(\mathbf{x}_{\bar{i}}) \leq 0, \quad \text{for each } i, 1 \leq i \leq n,$$

then there is an $x \in \mathbf{x}$ such that $f(x) = 0$.

Miranda’s theorem is a consequence of the Brouwer fixed point theorem, which we state on page 471 later.⁷

We now state our existence theorem for the interval Gauss–Seidel method.

THEOREM 8.11

Suppose $\mathbf{x}^{(1)}$ is defined through (8.62) (page 466) with $k = 0$, suppose that Y is nonsingular, and suppose that $\mathbf{x}^{(1)} \subseteq \mathbf{x}^{(0)}$. Then there is an $x \in \mathbf{x}^{(0)}$ such that $F(x) = 0$.

⁶We do not claim that this proof originated with us; these existence and uniqueness results are ubiquitous in the literature on interval Newton methods.

⁷The original presentation of Miranda’s theorem is in C. Miranda, “Un’ osservazione su un teorema di Brouwer,” *Bol. Un. Mat. Ital., Series 2*, pp. 5–7, 1940.

PROOF Using the multivariate mean value theorem and along the lines of (8.64), we have

$$(YF)_1(x_{\underline{1}}) = (YF)_1(\tilde{x}_1, \dots, \tilde{x}_n) + \sum_{\substack{j=1 \\ j \neq 1}}^n \left\{ \frac{\partial(YF)_1}{\partial x_j}(\underline{c})(x_j - \tilde{x}_j) \right\} \quad (8.65) \\ + \frac{\partial(YF)_1}{\partial x_j}(\underline{c})(\underline{x}_j - \tilde{x}_j),$$

where $\underline{c}$ is some point on the line between $(\underline{x}_1, \tilde{x}_2, \dots, \tilde{x}_n)^T$ and $(x_1, \dots, x_n)^T$, and

$$(YF)_1(x_{\overline{1}}) = (YF)_1(\tilde{x}_1, \dots, \tilde{x}_n) + \sum_{\substack{j=1 \\ j \neq 1}}^n \left\{ \frac{\partial(YF)_1}{\partial x_j}(\overline{c})(x_j - \tilde{x}_j) \right\} \quad (8.66) \\ + \frac{\partial(YF)_1}{\partial x_j}(\overline{c})(\overline{x}_j - \tilde{x}_j),$$

where $\overline{c}$ is some point on the line between $(\overline{x}_1, \tilde{x}_2, \dots, \tilde{x}_n)^T$ and $(x_1, \dots, x_n)^T$. Now, using the notation in the statement of Miranda's theorem,

$$(\mathbf{YF})_1^u(\mathbf{x}_{\underline{1}})(\mathbf{YF})_1^u(\mathbf{x}_{\overline{1}}) \leq 0 \quad (8.67)$$

if and only if

$$(YF)_1(x_{\underline{1}})(YF)_1(x_{\overline{1}}) \leq 0 \quad (8.68)$$

for every $x_{\underline{1}} \in \mathbf{x}_{\underline{1}}$ and every $x_{\overline{1}} \in \mathbf{x}_{\overline{1}}$.

Now observe that, in (8.65), $\partial(YF)_1/\partial x_j(\underline{c}) \neq 0$, since the denominator in the first step of the interval Gauss-Seidel method (8.62) is $(\partial \mathbf{YF})_1/\partial x_j(\mathbf{x})$, which contains $\partial(YF)_1/\partial x_j(\underline{c})$ by the fundamental theorem of interval arithmetic, and since there cannot be a zero in the denominator of (8.62) if the result is a bounded interval. Therefore,

$$\text{either } \frac{(\partial \mathbf{YF})_1}{\partial x_j}(\mathbf{x}) < 0 \quad \text{or} \quad \frac{(\partial \mathbf{YF})_1}{\partial x_j}(\mathbf{x}) > 0.$$

If $(\partial \mathbf{YF})_1/\partial x_j(\mathbf{x}) > 0$, then solving $(YF)_i(x_{\underline{1}}) \geq 0$ for $\underline{x}_1$ in (8.65) gives

$$\underline{x}_1 \leq \tilde{x}_1 - \frac{1}{\partial(YF)_1/\partial x_j(\underline{c})} \left\{ (YF)_1(\tilde{x}_1, \dots, \tilde{x}_n) + \sum_{\substack{j=1 \\ j \neq 1}}^n \left\{ \frac{\partial(YF)_1}{\partial x_j}(\underline{c})(x_j - \tilde{x}_j) \right\} \right\}. \quad (8.69)$$

However, the right member of (8.69) is contained in $\mathbf{x}_i^{(1)}$, and the lower bound of $\mathbf{x}_i^{(1)}$ is greater than $\underline{x}_1$ by the hypothesis to the theorem; therefore, (8.69) is true, and $(\mathbf{YF})_1^u(\mathbf{x}_{\underline{1}}) \geq 0$. Similarly, in the same case $(\partial \mathbf{YF})_1/\partial x_j(\mathbf{x}) > 0$ we can use (8.66), the fundamental theorem of interval arithmetic, and the

hypothesis of the theorem to show that $(\mathbf{YF})_1^u(\mathbf{x}_1^-) \leq 0$. Thus, the part of the hypothesis to Miranda's theorem corresponding to $i = 1$ holds when $(\partial \mathbf{YF})_1 / \partial x_j(\mathbf{x}) > 0$.

A similar argument holds for $i = 1$ when $(\partial \mathbf{YF})_1 / \partial x_j(\mathbf{x}) < 0$. You will fill in the details in Exercise 22 on page 485.

For $i = 2$, the box $\mathbf{x}^{(0)}$ must be replaced by a smaller box contained in $\mathbf{x}^{(0)}$, and the above argument holds over this sub-box. We then repeat the process for $i = 3, \dots, n$. Since the hypotheses of Miranda's theorem then hold for $(\mathbf{YF})$ over some sub-box of $\mathbf{x}$, it follows that there is a solution of $(\mathbf{YF})(\mathbf{x}) = 0$ in this sub-box, and hence in $\mathbf{x}$.

To complete the proof, we observe that, if $\mathbf{Y}$ is nonsingular, then the only solution to $\mathbf{Y}\mathbf{v} = 0$ is $\mathbf{v} = 0$; therefore, if $\mathbf{YF} = 0$, it follows that $\mathbf{F} = 0$. $\square$

REMARK 8.13 More powerful existence results can be proven for the interval Gauss–Seidel method; see, for example, the results in [62]. However, we have presented the above theorem since it is relatively easy to prove, gives a result that is useful in practice, and for which it is easy to see the main idea. $\square$

8.4.2 The Multivariate Krawczyk Method

The multivariate Krawczyk method is a tight analogue of the univariate Krawczyk method, introduced in Problem 29 on page 81. In particular, we set

$$G(\mathbf{x}) = \mathbf{x} - \mathbf{YF}(\mathbf{x}), \quad (8.70)$$

where $\mathbf{Y}$ is an approximation to $(\mathbf{F}'(\tilde{\mathbf{x}}))^{-1}$, for some point $\tilde{\mathbf{x}}$ near $\mathbf{x}$. In matrix form,

$$G'(\mathbf{x}) = \mathbf{I} - \mathbf{YF}'(\mathbf{x}). \quad (8.71)$$

(See Problem 23 on page 485). Applying our multivariate mean value theorem (Theorem 8.1 on page 441) to (8.70), then replacing the matrix $\mathbf{A}$ in (8.1) by a matrix $\mathbf{F}'(\mathbf{x})$ whose entries represent interval enclosures for the range of corresponding entries of $\mathbf{F}'$ over $\mathbf{x}$, we thus obtain

$$\begin{aligned} G(\mathbf{x}) &\in G(\tilde{\mathbf{x}}) + (\mathbf{I} - \mathbf{YF}'(\mathbf{x}))(\mathbf{x} - \tilde{\mathbf{x}}) \\ &= \tilde{\mathbf{x}} - \mathbf{YF}(\tilde{\mathbf{x}}) + (\mathbf{I} - \mathbf{YF}'(\mathbf{x}))(\mathbf{x} - \tilde{\mathbf{x}}). \end{aligned} \quad (8.72)$$

This leads us to

DEFINITION 8.11 (*The multivariate Krawczyk operator*)

$$\mathbf{x}^{(k+1)} = \mathbf{K}(\mathbf{F}, \mathbf{x}^{(k)}, \tilde{\mathbf{x}}^{(k)}) = \tilde{\mathbf{x}}^{(k)} - \mathbf{YF}(\tilde{\mathbf{x}}^{(k)}) + (\mathbf{I} - \mathbf{YF}'(\mathbf{x}^{(k)}))(\mathbf{x}^{(k)} - \tilde{\mathbf{x}}^{(k)}) \quad (8.73)$$

is called the Krawczyk method, where the operator $\mathbf{K}(F, \mathbf{x}^{(k)}, \tilde{\mathbf{x}}^{(k)})$ is called the Krawczyk operator.

REMARK 8.14 The point $\tilde{\mathbf{x}}^{(k)}$ is often chosen to be the vector of midpoints of components of $\mathbf{x}^{(k)}$, while Y is often chosen to be the inverse of the matrix whose entries are the midpoints of corresponding entries of $\mathbf{F}'(\mathbf{x})$. These choices endow the Krawczyk method with certain symmetries and nice theoretical properties. $\square$

We have

THEOREM 8.12

Suppose $\mathbf{K}(F, \mathbf{x}^{(k)}, \tilde{\mathbf{x}}^{(k)}) \subset \mathbf{x}^{(k)}$ and Y is nonsingular. Then there exists an $x \in \mathbf{x}$ such that $F(x) = 0$.

REMARK 8.15 The condition Y can be removed, but the proof of the theorem is slightly simpler if we make this assumption. $\square$

This theorem is most easily proven with

THEOREM 8.13

(The Brouwer fixed point theorem) Suppose D is a closed, bounded, convex set, suppose $G : D \rightarrow \mathbb{R}^n$, where G is continuous, and suppose $G(x) \in D$ for every $x \in D$. Then there is an $\mathbf{x}^* \in D$ such that $G(\mathbf{x}^*) = \mathbf{x}^*$, that is, G has a fixed point in D .

REMARK 8.16 With introduction of additional terminology from algebraic topology, the Brouwer fixed point theorem can be stated somewhat more generally, but this statement is sufficient for our purposes. $\square$

REMARK 8.17 The Brouwer fixed point theorem is due to Felix Brouwer, one of the fathers of algebraic topology, in 1909, but is now common knowledge and is widely used among mathematical economists, analysts, etc. $\square$

REMARK 8.18 The Brouwer fixed point theorem is a partial strengthening of the contraction mapping theorem (Theorem 8.2 on page 442). In particular, one of the assumptions in the contraction mapping theorem is that G map D into itself. That is the only assumption in the Brouwer fixed point theorem. However, the conclusion of the Brouwer fixed point theorem is somewhat weaker, since the Brouwer fixed point theorem doesn't mention

an iteratively defined sequence that converges to the fixed point, nor does it claim that the fixed point is unique. $\square$

Now, we prove Theorem 8.12.

PROOF Equation (8.72), the definition of $\mathbf{K}(F, \mathbf{x}^{(k)}, \tilde{x}^{(k)})$, and the fundamental theorem of interval arithmetic (Theorem 1.9 on page 26) imply $G(x) \in \mathbf{K}(F, \mathbf{x}^{(k)}, \tilde{x}^{(k)})$ for every $x \in \mathbf{x}$. Combining this with the hypothesis of Theorem 8.12 then gives $G(x) \in \mathbf{x}$ for $x \in \mathbf{x}$. The Brouwer fixed point theorem therefore implies that G has a fixed point $\mathbf{x}^* \in \mathbf{x}$, $G(\mathbf{x}^*) = \mathbf{x}^*$. However, this implies that $YF(x) = 0$. The conclusion of Theorem 8.12 now follows from the nonsingularity of Y . $\square$

The condition $\mathbf{K}(F, \mathbf{x}^{(k)}, \tilde{x}^{(k)}) \subset \mathbf{x}$ can be combined with other conditions, such as $\|I - YF'(\mathbf{x}^{(k)})\| \leq r < 1$, to imply that Krawczyk iteration converges to a small interval bounding the unique fixed point.⁸

8.4.3 Using Interval Gaussian Elimination

Interval Gaussian elimination, as we saw in Section 3.3.7 (page 130), can be used to bound the solution set to the interval linear system (8.60) (on page 464) in the general multivariate interval Newton operator. Under certain circumstances, interval Gaussian elimination gives better results than the Krawczyk method and the interval Gauss–Seidel method, but the interval Gauss–Seidel method gives better results in other situations. See A. Neumaier, *Interval Methods for Nonlinear Systems*, Cambridge University Press, 1990 for details.

8.4.4 Relationship to the Kantorovich Theorem

The Kantorovich theorem (Theorem 8.7 on page 454) has conclusions that are similar to the conclusions for the existence and uniqueness theorems for interval Newton methods. In particular, if a certain set is bounded inside another set, then existence of a solution within a particular region is assured. There are several papers in the numerical analysis literature, such as [72] (1980) and [64]. In the latter, it is shown that, if slopes are used instead of Lipschitz sets, the existence test based on the Krawczyk method gives results whenever the Kantorovich theorem does, and sometimes gives results when the Kantorovich theorem does not.

⁸For details, see A. Neumaier, *Interval Methods for Nonlinear Systems*, Cambridge University Press, 1990, R. E. Moore, “A Test for Existence of Solutions to Nonlinear Systems, *SIAM J. Numer. Anal.* **14** (4), pp. 611–615 (September, 1977), etc.

8.5 Quasi-Newton Methods (Broyden's Method)

Many point iterative methods for finding solutions to $F(x) = 0$ have the form

$$\begin{cases} p^{(k)} &= -H^{(k)}F(x^{(k)}) \\ x^{(k+1)} &= x^{(k)} + p^{(k)}t^{(k)} \end{cases} \text{ for } k = 0, 1, 2, \dots, \quad (8.74)$$

where $H^{(k)}$ is an $n \times n$ matrix and $t^{(k)}$ is a scalar. For example, if $H^{(k)} = (F'(x^{(k)}))^{-1}$ and $t^{(k)} = 1$, (8.74) defines Newton's method.

However, $H^{(k)}$ may be chosen to satisfy certain conditions such that some properties of $H^{(k)}$ approximate those of $(F'(x^{(k)}))^{-1}$. Such methods, called *quasi-Newton methods*, are regarded as variations of Newton's method. We will study a particular quasi-Newton method called Broyden's method [25]. We will see that Broyden's method reduces by an order of magnitude the $\mathcal{O}(n^2)$ scalar function evaluations and the $\mathcal{O}(n^3)$ operations involved in solving the linear system at each iteration of Newton's method.

We now derive Broyden's method. We assume:

(a) F is continuously differentiable on an open set $D \subset \mathbb{R}^n$.

(b) For given $x \in D$ and given $s \neq 0$, $x^+ = x + s \in D$.

We associate x with $x^{(k)}$ and x^+ with $x^{(k+1)}$, and we seek a good approximation to $F'(x^{(k+1)})$. Since F' is continuous at x^+ , given $\epsilon > 0$ there is a $\delta > 0$ such that

$$\|F(x) - F(x^+) - F'(x^+)(x - x^+)\| \leq \epsilon \|x - x^+\|,$$

provided $\|x - x^+\| < \delta$. It follows that

$$F(x) \approx F(x^+) + F'(x^+)(x - x^+),$$

with the approximation improving as $\|x - x^+\|$ decreases. Hence, if B^+ is to denote an approximation to matrix $F'(x^+)$, it is natural to require that B^+ satisfy

$$F(x) = F(x^+) + B^+(x - x^+),$$

that is,

$$B^+s = y = F(x^+) - F(x), \quad \text{where } s = x^+ - x. \quad (8.75)$$

Equation (8.75), called the *secant equation* or *quasi-Newton equation*, is central to the development of quasi-Newton methods.

REMARK 8.19 If $n = 1$, (8.75) completely determines B^+ , and we are led to the secant method, i.e.

$$f'(x^{(k+1)}) \approx \frac{f(x^{(k+1)}) - f(x^{(k)})}{x^{(k+1)} - x^{(k)}} = B^+. \quad (8.76)$$

(You will show this in Exercise 24 on page 485.) □

For $n > 1$, the quasi-Newton equation deals with the approximate change in $F(x)$ in the direction $s = x^+ - x$. Now suppose that we have an approximation B to $F'(x)$, i.e., $B \simeq F'(x)$. Broyden assumed that $B^+ \approx F'(x^+)$ approximately produces the same effect as B in any direction orthogonal to s . Thus, we assume that

$$B^+z = Bz \quad \text{if} \quad z^Ts = 0. \quad (8.77)$$

It turns out that Equations (8.75) and (8.77) uniquely determine B^+ from B . To find B^+ , consider the matrix

$$A = \frac{(y - Bs)s^T}{s^Ts}, \quad \text{with } y = B^+s \text{ from (8.75).}$$

Let $v \in \mathbb{R}^n$. Then $v = z + as$ for some scalar a , because

$$\mathbb{R}^n = \text{span}(s, z_1, z_2, \dots, z_{n-1}),$$

where $z_1, z_2, \dots, z_{n-1}$ are orthogonal to s and $z = \sum_{i=1}^{n-1} c_i z_i$. Then,

$$\begin{aligned} Av = A(z + as) &= \frac{(y - Bs)s^T(z + as)}{s^Ts} \\ &= a(B^+s - Bs) \quad (\text{since } s^Tz = 0) \\ &= B^+(z + as) - B(z + as) \\ &= (B^+ - B)v, \end{aligned}$$

since $Av = (B^+ - B)v$ for all $v \in \mathbb{R}^n$, $A = B^+ - B$. Thus,

$$B^+ = B + \frac{(y - Bs)s^T}{s^Ts}. \quad (8.78)$$

Equation (8.78) provides what is known as the *Broyden update* to the approximate Jacobian matrix.

An alternative way of viewing the above construction of B^+ from B is to view

$$B^+ = B + v \frac{1}{s^Ts} s^T, \quad v \text{ an arbitrary vector}$$

as a perturbation of B by the rank-one matrix $vs^T/(s^Ts)$ such that $B^+s = Bs + v$, but $B^+w = Bw$ for every w with $s^Tw = 0$. We then see that, for $B^+s = y$, we must have $v = y - Bs$.

Another way to see that (8.78) is a good choice for B^+ (subject to the condition that B^+ satisfies (8.75)), is that B^+ given by (8.78) is the “closest” matrix to B in the Euclidean norm from all matrices that satisfy (8.75). This is stated in the following proposition.

PROPOSITION 8.4

Given B an $n \times n$ matrix and $y \in \mathbb{R}^n$ and some nonzero $s \in \mathbb{R}^n$, define B^+ by (8.78). Then B^+ is the unique solution to the problem

$$\min\{\|\hat{B} - B\|_E : \hat{B}s = y\}, \quad \text{where} \quad \|A\|_E^2 = \sum_{i,j=1}^n |a_{ij}|^2.$$

PROOF To show that B^+ is a solution, note that if $y = \hat{B}s$, then

$$\|B^+ - B\|_E = \left\| (\hat{B} - B) \frac{1}{s^T s} s s^T \right\|_E \leq \|\hat{B} - B\|_E \frac{\|s s^T\|_E}{s^T s} = \|\hat{B} - B\|_E \quad (8.79)$$

(from (8.78), defining the Broyden update). That B^+ is a unique solution follows from the following argument: Suppose that B_1 and B_2 are two solutions and $B_1 \neq B_2$, i.e.,

$$\|B_1 - B\|_E \leq \|\hat{B} - B\|_E \quad \text{and} \quad B_1 s = y,$$

and

$$\|B_2 - B\|_E \leq \|\hat{B} - B\|_E \quad \text{and} \quad B_2 s = y$$

for every $\hat{B}$ that satisfies $\hat{B}s = y$. Let $B^* = \lambda B_1 + (1 - \lambda)B_2$, where λ is any number with $0 < \lambda < 1$. Then

$$B^* s = y$$

and

$$\begin{aligned} \|B^* - B\|_E &= \|\lambda(B_1 - B) + (1 - \lambda)(B_2 - B)\|_E \\ &< \lambda\|(B_1 - B)\|_E + (1 - \lambda)\|(B_2 - B)\|_E \leq \|\hat{B} - B\|_E. \end{aligned} \quad (8.80)$$

(The proof of the first strict inequality depends on the fact that $(B_1 - B) \neq \lambda(B_2 - B)$ for any λ ; see Remark 8.20 below.) Thus, $\|B^* - B\|_E < \|\hat{B} - B\|_E$, which is a contradiction when $\hat{B} = B^*$. Thus, B^+ is unique. $\square$

REMARK 8.20 In general, if A and B are $n \times n$ ($n > 1$) nonzero matrices, $A \neq \alpha B$ for any scalar α , and $0 < \lambda < 1$, then

$$\|\lambda A + (1 - \lambda)B\|_E < \lambda\|A\|_E + (1 - \lambda)\|B\|_E.$$

We say that the Euclidean norm is *strictly convex*. (This is to be shown in Exercise 25 on page 485.) $\square$

We now consider how (8.75) and (8.78) can be used in an iterative method to solve $F(x) = 0$. The basic formulas for Broyden's method are

$$x^{(k+1)} = x^{(k)} - B_k^{-1} F(x^{(k)}), \quad k = 0, 1, 2, \dots \quad (8.81)$$

$$y^{(k)} = F(x^{(k+1)}) - F(x^{(k)}), \quad s^{(k)} = x^{(k+1)} - x^{(k)} \quad (8.82)$$

$$B_{k+1} = B_k + \frac{(y^{(k)} - B_k s^{(k)}) s^{(k)T}}{s^{(k)T} s^{(k)}}. \quad (8.83)$$

It is clear that, given $x^{(0)}$ and B_0 (either $F'(x^{(0)})$ or a good approximation to $F'(x^{(0)})$), Broyden's method can be carried out with n scalar function evaluations per time step, i.e., Broyden's method requires only evaluation of $F(x^{(k+1)})$ (and no partial derivatives) in each step. However, it appears that we still need to solve a linear system $B_k s^{(k)} = -F(x^{(k)})$ at each time step. We can overcome this difficulty by using a result of Sherman and Morrison. First, we need the following lemma.

LEMMA 8.5

Let $v, w \in \mathbb{R}^n$ be given. Then

$$\det(I + vw^T) = 1 + w^T v. \quad (8.84)$$

PROOF Let $P = I + vw^T$. If $v = 0$, the result is trivial, so assume that $v \neq 0$. Let z be an eigenvector of P , i.e., $(I + vw^T)z = \lambda z$ for some λ ; then $(1 - \lambda)z = -(w^T z)v$, i.e., z is either orthogonal to w or is a multiple of v . (Thus, $n - 1$ eigenvectors are orthogonal to w .) If $w^T z = 0$, then $\lambda = 1$, while if z is parallel to v , $\lambda = 1 + w^T v$; thus, the eigenvalues of P are all 1 except for a *single* eigenvalue equal to $1 + w^T v$. Thus, (8.84) follows from

$$\det(P) = \prod_{i=1}^n \lambda_i = 1 + w^T v.$$

□

LEMMA 8.6

(The Sherman–Morrison formula) Let $u, v \in \mathbb{R}^n$ and assume that the $n \times n$ matrix A is nonsingular. Then $A + uv^T$ is nonsingular if and only if $\sigma = 1 + v^T A^{-1}u \neq 0$. Moreover, if $\sigma \neq 0$, then

$$(A + uv^T)^{-1} = A^{-1} - \frac{1}{\sigma} A^{-1} uv^T A^{-1}. \quad (8.85)$$

PROOF Since $\det(A + uv^T) = \det(A) \det(I + A^{-1}uv^T)$ and A is nonsingular, $A + uv^T$ is nonsingular if and only if $\det(I + A^{-1}uv^T) \neq 0$. By Lemma 8.5, $\det(I + A^{-1}uv^T) = 1 + v^T A^{-1}u = \sigma$. To verify (8.85), we need

only show that if we multiply the right-hand side by $A + uv^T$ we get I . Thus,

$$\begin{aligned}
 & (A + uv^T)(A^{-1} - \frac{1}{\sigma}A^{-1}uv^TA^{-1}) \\
 &= I + uv^TA^{-1} - \frac{1}{\sigma}(uv^TA^{-1} + uv^TA^{-1}uv^TA^{-1}) \\
 &= I - \frac{1}{\sigma}[-\sigma uv^TA^{-1} + uv^TA^{-1} + u(v^TA^{-1}u)v^TA^{-1}] \\
 &= I - \frac{1}{\sigma}[-uv^TA^{-1} - v^TA^{-1}uuv^TA^{-1} \\
 &\quad + uv^TA^{-1} + (v^TA^{-1}u)uv^TA^{-1}] \\
 &= I
 \end{aligned}$$

□

Now consider the iterative procedure (8.81)–(8.82) in light of Lemma 8.2. We have

$$B_{k+1} = B_k + \frac{(y^{(k)} - B_k s^{(k)})s^{(k)T}}{s^{(k)T}s^{(k)}},$$

which has the form

$$B_{k+1} = B_k + uv^T, \quad \text{where } u = \frac{y^{(k)} - B_k s^{(k)}}{s^{(k)T}s^{(k)}} \text{ and } v^T = s^{(k)T}.$$

Thus, by Lemma 8.6 (the Sherman–Morrison formula),

$$B_{k+1}^{-1} = (B_k + uv^T)^{-1} = B_k^{-1} - \frac{1}{\sigma}B_k^{-1}uv^TB_k^{-1}, \quad \text{where } \sigma = 1 + v^TB_k^{-1}u.$$

Thus,

$$(B_{k+1})^{-1} = B_k^{-1} - \frac{\frac{1}{s^{(k)T}s^{(k)}}(B_k^{-1}y^{(k)} - s^{(k)})s^{(k)T}B_k^{-1}}{1 + s^{(k)T}\left[\frac{B_k^{-1}y^{(k)} - s^{(k)}}{s^{(k)T}s^{(k)}}\right]}.$$

Hence,

$$(B_{k+1}^{-1}) = B_k^{-1} + \frac{(s^k - B_k^{-1}y^{(k)})s^{(k)T}B_k^{-1}}{s^{(k)T}B_k^{-1}y^{(k)}}.$$

Letting $H_k = B_k^{-1}$ and $H_{k+1} = B_{k+1}^{-1}$, we have

$$H_{k+1} = H_k + \frac{(s^k - H_k y^{(k)})s^{(k)T}H_k}{s^{(k)T}H_k y^{(k)}}. \quad (8.86)$$

Therefore, Broyden's Method can be implemented in the following manner:

$$x^{(k+1)} = x^{(k)} - H_k F(x^{(k)}), \quad k = 0, 1, 2, \dots \quad (8.87)$$

$$y^{(k)} = F(x^{(k+1)}) - F(x^{(k)}), \quad s^{(k)} = x^{(k+1)} - x^{(k)} \quad (8.88)$$

$$H_{k+1} = H_k + \frac{(s^{(k)} - H_k y^{(k)}) s^{(k)T} H_k}{s^{(k)T} H_k y^{(k)}}. \quad (8.89)$$

Here, $x^{(0)}$ is an initial guess, and $H_0 = (F'(x^{(0)}))^{-1}$ or H_0 is a good approximation to $(F'(x^{(0)}))^{-1}$. In the above form, Broyden's Method only requires n scalar function evaluations, i.e., $F(x^{(k)})$, and $\mathcal{O}(n^2)$ arithmetic operations per iteration, i.e., the matrix-vector multiplications involving H_k .

REMARK 8.21 One problem with Broyden's Method is that B_{k+1} may be singular for some k ; in such cases $s^{(k)T} H_k y^{(k)} = 0$ in (8.89). Broyden's Method is then sometimes implemented in the following form rather than in the form (8.78):

$$B^+ = B + \theta \frac{(y - Bs)s^T}{s^T s}, \quad (8.90)$$

where θ is chosen to avoid a singular B^+ . (Note that $\theta = 1$ in (8.78).) To avoid singular B^+ , Lemma 8.1 and Formula (8.90) are used to yield

$$\det B^+ = \det(B) \times [(1 - \theta) + \theta \frac{s^T B^{-1} y}{s^T s}], \quad (8.91)$$

$$(\text{since } B^+ = B \left[I + \theta \frac{(B^{-1}y - s)s^T}{s^T s} \right]). \quad \square$$

Now θ is chosen as close to 1 as possible subject to $|\det B^+| > \sigma |\det(B)|$ for some specified $\sigma \in (0, 1)$.

We have the following local convergence theorem for Broyden's method, which we present without proof.

THEOREM 8.14

Let F be continuously differentiable on an open convex set $D \in \mathbb{R}^n$. Let there be an $x^* \in D$ such that $F(x^*) = 0$, and $F'(x^*)$ is nonsingular. Furthermore, suppose that there is a constant $\hat{c}$ such that

$$\|F'(x) - F'(x^*)\| \leq \hat{c} \|x - x^*\| \quad \text{for } x \in D.$$

Then, Broyden's Method is locally and superlinearly convergent to x^* .

REMARK 8.22 Theorem 8.14 states that

$$\frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} \leq \alpha_k,$$

where $\alpha_k \rightarrow 0$ as $k \rightarrow \infty$. Under the same hypotheses, Newton's method is quadratically convergent, i.e.,

$$\frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|^2} \leq c$$

for k sufficiently large. (See Proposition 8.1 on page 450.) $\square$

REMARK 8.23 Although $x^{(k)} \rightarrow x^*$ superlinearly, it is not necessarily true that $B_k \rightarrow F'(x^*)$ as $k \rightarrow \infty$. (Indeed, Broyden's Method is not self-correcting, i.e., B_k may retain harmful information contained in B_j , $j < k$.) Consider

$$\begin{aligned} f_1(x_1, x_2) &= x_1, \\ f_2(x_1, x_2) &= x_2 + x_2^3, \end{aligned}$$

where

$$F'(x) = \begin{pmatrix} 1 & 0 \\ 0 & 1 + 3x_2^2 \end{pmatrix}, \quad F'(x^*) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \text{and} \quad B_0 = \begin{pmatrix} 1 + \delta & 0 \\ 0 & 1 \end{pmatrix}.$$

However, the $(1,1)$ element of B_k can be shown to be $1 + \delta$ for all k ; thus, $\{B_k\}$ does not converge to $F'(x^*)$. $\square$

8.5.1 Practicalities

Quasi-Newton methods were originally developed in an era when finding a solution to 10 nonlinear equations in 10 variables was a challenge for many systems that are almost trivial today, when symbolic computation (computer algebra) was in its infancy, etc. One of the original rationales was to avoid having to derive and program a Jacobian matrix. For many systems today, automatic differentiation (as introduced in Section 6.2, page 327) is practical, and is used, due to the superior convergence properties of Newton's method; furthermore, automatic differentiation can compute matrix-vector multiples $F'(x)v$ in less than $\mathcal{O}(n^2)$ operations.

However, quasi-Newton methods with "secant updates" (updates to the matrix B_k defined by the secant equation (8.75)) are still useful in modern scientific computation, such as for truly *black box systems*⁹ or for very large systems where a structure can be imposed on the B_k , etc.

⁹that is, for functions F whose values are obtained by some process which cannot be analyzed. Such a process is called a "black box."

8.6 Methods for Finding All Solutions

To this point, we have discussed iterative methods for finding approximations to a single solution to a nonlinear system of equations. In many applications, finding all solutions to a nonlinear system is required. Salient among these are *homotopy methods* and *branch and bound methods*.

8.6.1 Homotopy Methods

In a homotopy method, one starts with a simple function $g(x)$, $g : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ such that every point with $g(x) = 0$ is known, then transforms the function into the $f(x)$, $f : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ for which all points satisfying $f(x) = 0$ are desired. During the process, one solves various intermediate systems, using the solution to the previous system in an initial guess for an iterative method for the next system. A typical such transformation is

$$H(x, t) = (1 - t)f(x) + tg(x), \quad (8.92)$$

so $H(x, 0) = f(x)$ and $H(x, 1) = g(x)$. One way of following the curves $H(x, t) = 0$ from $t = 0$ to $t = 1$ is to consider $y = (x, t) \in \mathbb{R}^{n+1}$, and to differentiate (8.92), obtaining

$$H'(y)y' = 0, \quad (8.93)$$

where $H'(z)$ is the n by $n+1$ Jacobian matrix of H . If H' is of full rank, $H'(z)$ has a one-dimensional null space, parallel to y' . At step k of the method, one can take a vector v_k in this direction, with $H'(y_k)v_k = 0$, say $\|v\| = 1$, and say $v_k^T v_{k-1} > 0$. One computes a *predictor step*

$$z_k = y_k + \epsilon_k v_k, \quad (8.94)$$

then one corrects z_k by iterating Newton's method on the system

$$\begin{pmatrix} H(y) \\ N(y) \end{pmatrix} = 0, \quad (8.95)$$

where $N(y)$ is a normalization function. For example, if we want to correct in a direction perpendicular to the predictor step v_k , we could use

$$N(y) = v_k^T (y - z_k). \quad (8.96)$$

Similarly, if t corresponds to the $(n+1)$ -st coordinate of y and we want to correct perpendicularly to that direction, we could set $N(y)$ to be the difference between the $(n+1)$ -st coordinates of y and z_k . Such a two-step approach (computing z_k tangent to the curve, then correcting back onto the curve) is called

a *predictor-corrector method*, not to be confused with a predictor-corrector method for differential equations.

In fact, however, (8.92) along with the normalization condition defines a derivative y' , so, in principle, methods and software for finding solutions to initial value problems for ordinary differential equations can be used to follow the curves of the homotopy. Indeed, this approach has been used.

Determining an appropriate starting function g is a crucial part of a homotopy method for finding all solutions to a system of equations. Particularly interesting is finding such g for polynomial systems of equations, where there is an interplay between numerical analysis and algebraic geometry. Significant results were obtained during the 1980's; for example, see [59]. In such techniques, the homotopy is generally defined in a space derived from complex n -space, rather than real n -space.

While finding all solutions to a nonlinear system with these techniques is called a homotopy method, an actual technique for following the solution curves of a system of equations $H(y) = 0$, where $H : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^n$ is termed a *continuation method*. In addition to solving systems of nonlinear equations in homotopy methods, continuation methods are used to analyze parameterized systems of differential equations as the parameter (which we can view for now as the variable t) changes. In such systems, points along solution curves at which the Jacobian matrix H' is not of full rank are of interest. Those points, where two or more solution curves can intersect, are termed *bifurcation points*. Bifurcation points are of physical significance in the models giving rise to these systems,¹⁰ and there is a rich mathematical theory for classification and analysis of bifurcation points.

An introduction to continuation methods is [2], while an example of software is [98]. A relatively early reference on use of homotopy methods for solving polynomial systems is [60]; search the web for more recent work.

8.6.2 Branch and Bound Methods

Branch and bound methods, which we explain later in §9.6.3 in the context of optimization, can also be used to solve systems of nonlinear equations. In this context, the equations $F(x) = 0$ can be considered as constraints, and the objective function can be $\sum_{i=1}^n f_i^2(x)$, for example. See §9.6.3 for clarification.

¹⁰in fluid dynamics, biology, economics, etc.

8.7 Exercises

1. Use the univariate mean value theorem (which you can find stated as Theorem 1.4 on page 3) to prove the multivariate mean value theorem (stated as Theorem 8.1 on page 441).
2. Write down the degree 2 Taylor polynomials for $f_1(x_1, x_2)$ and $f_2(x_1, x_2)$, centered at $\tilde{x} = (\tilde{x}_1, \tilde{x}_2) = (0, 0)$, for F as in Example 8.1. Lumping terms together in an appropriate way, interpret your values in terms of the Jacobian matrix and a second-derivative tensor.
3. Show that if F is Fréchet differentiable at x , then F is continuous at x , i.e., prove that given $\epsilon > 0$ there is a $\delta > 0$ such that $\|F(x) - F(y)\| < \epsilon$ whenever $\|x - y\| < \delta$.
4. Let F be as in Example 8.1 (on page 441), and define

$$G(x) = x - YF(x), \quad \text{where } Y \approx \begin{pmatrix} 2.0030 & 1.2399 \\ 0.0262 & 0.0767 \end{pmatrix}.$$

- (a) Do several iterations of fixed point iteration, starting with initial guess $x^{(0)} = (8.0, -0.9)^T$. What do you observe?
 - (b) Use Theorem 8.3 (on page 444) and Theorem 8.2 (the Contraction Mapping Theorem, page 442), if possible, to show that fixed point iteration will converge within a ball of radius 0.001 of $x = (-8.2005, -0.8855)^T$. Relate this to Theorem 8.4.
5. The nonlinear system

$$\begin{aligned} x_1^2 - 10x_1 + x_2^2 + 8 &= 0, \\ x_1x_2^2 + x_1 - 10x_2 + 8 &= 0 \end{aligned}$$

can be transformed into the fixed-point problem

$$\begin{aligned} x_1 &= g_1(x_1, x_2) = \frac{x_1^2 + x_2^2 + 8}{10}, \\ x_2 &= g_2(x_1, x_2) = \frac{x_1x_2^2 + x_1 + 8}{10}. \end{aligned}$$

Show that $G(x) = (g_1(x), g_2(x))^T$ has a unique fixed point in

$$D_0 = \{(x_1, x_2) \in \mathbb{R}^2 : 0 \leq x_1, x_2 \leq 1.5\},$$

and that the fixed point iterations $x^{(k+1)} = G(x^{(k)})$ converge for any $x^{(0)} \in D_0$.

6. Perform 4 iterations of the fixed-point method in problem 5 with initial vector $x^{(0)} = (0.5, 0.5)^T$.
7. Let B be an $n \times n$ real matrix with $\rho(B) < 1$ and define $G : \mathbb{R}^n \rightarrow \mathbb{R}^n$ by $G(x) = Bx + b$ with $b \in \mathbb{R}^n$. Show that G has a unique fixed point x^* and the iterates $x^{(k+1)} = G(x^{(k)})$ converge to x^* for any $x^{(0)} \in \mathbb{R}^n$.

8. Consider $x^{(k+1)} = \begin{bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \end{bmatrix} = \begin{bmatrix} 0.5 & -0.25 \\ -0.25 & 0.5 \end{bmatrix} \begin{bmatrix} \cos(x_1^{(k)}) \\ \sin(x_2^{(k)}) \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix}$,
with $x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

- (a) Prove that $\{x^{(k)}\}_{k=0}^{\infty}$ converges to a unique $x^* \in \mathbb{R}^2$.
- (b) Estimate the number of iterations required to achieve

$$\|x^{(k)} - x^*\|_{\infty} < 0.001.$$

9. Univariate Newton iteration applied to find complex roots

$$f(x + iy) = u(x, y) + iv(x, y) = 0$$

is equivalent to multivariate Newton iteration with functions

$$\begin{aligned} f_1(x, y) &= u(x, y) = 0 \quad \text{and} \\ f_2(x, y) &= v(x, y) = 0. \end{aligned}$$

- (a) Repeat Exercise 35 on page 82 in Section 2.8, except doing the iterations on the corresponding system $u(x, y) = 0$, $v(x, y) = 0$ of two equations in two unknowns.
- (b) Compare the results, number by number, to the results you obtained in Exercise 35 on page 82 in Section 2.8.
10. Apply several iterations of Equation (8.37) (on page 453), for computing the inverse of the matrix

$$A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix},$$

with starting matrix $x^{(0)}$ equal to the diagonal matrix whose diagonal entries are $1/2$.

- (a) Do you observe quadratic convergence?
- (b) How many operations per iteration are needed if A is a general n by n matrix? How many are needed if A is an n by n tridiagonal matrix?

- (c) Do you see situations where this method of computing the inverse of a matrix might be more practical than Gaussian elimination?
11. Show that the one-step SOR–Newton method has the form in Equation (8.58).
12. Consider solving the nonlinear system

$$\begin{aligned}x_1^2 - 10x_1 + x_2^2 + 8 &= 0, \\x_1x_2^2 + x_1 - 10x_2 + 8 &= 0.\end{aligned}$$

Perform 4 iterations of Newton's method with the initial vector $x^{(0)} = (0.5, 0.5)^T$.

13. If $F(x) = (f_1(x), f_2(x), \dots, f_n(x))$, where

$$\begin{aligned}f_1(x) &= x_1 - a \\f_i(x) &= -x_{i-1} + 3x_i + e^{x_i} - x_{i+1} \text{ for } i = 2, 3, \dots, n-1, \text{ and} \\f_n(x) &= x_n - b,\end{aligned}$$

then $F(x) = 0$ has a unique solution $x^* \in \mathbb{R}^n$. Show that:

- (a) F is continuously F -differentiable for all $x \in \mathbb{R}^n$.
 (b) F is convex on all of $\mathbb{R}^n$.
 (c) $(F'(x))^{-1}$ exists for all $x \in \mathbb{R}^n$.
 (d) $(F'(x))^{-1} \geq 0$ for all $x \in \mathbb{R}^n$.

Moreover, show that for any $x^{(0)} \in \mathbb{R}^n$, the Newton iterates converge to x^* .

14. Consider finding the minimum of

$$f(x_1, x_2) = e^{x_1} + e^{x_2} - x_1x_2 + x_1^2 + x_2^2 - x_1 - x_2 + 4$$

on $\mathbb{R}^2$. Prove that Newton's method

$$x^{(k+1)} = x^{(k)} - \left(\nabla^2 f(x^{(k)}) \right)^{-1} \nabla f(x^{(k)})$$

converges to the unique minimum $x \in \mathbb{R}^2$ for any initial guess $x^{(0)} \in \mathbb{R}^2$.

15. Show that, if $\mathbf{A}$ is a Lipschitz matrix for F over $\mathbf{x}$, then $\mathbf{A}$ is a slope matrix for F at $\tilde{x}$ over $\mathbf{x}$, for any $\tilde{x} \in \mathbf{x}$.
16. Suppose $F : \mathbf{x} \rightarrow \mathbb{R}^n$, where $\mathbf{x}$ is an n -dimensional vector whose entries are intervals, and suppose we form an interval matrix $\mathbf{A}$ such that the (i, j) -th entry of $\mathbf{A}$ is an interval extension of the j -th partial derivative $\partial f_i / \partial x_j$ of f_i over $\mathbf{x}$. Show that $\mathbf{A}$ is a Lipschitz matrix for F over $\mathbf{x}$.

17. Prove Proposition 8.2 (on page 464).
18. Let F be as in Exercise 9 on page 483, let $\mathbf{x} = ([-0.1, 0.2], [0.8, 1.1])^T$, and $\tilde{\mathbf{x}} = (0.05, 0.95)^T$.
 - (a) Apply several iterations of the interval Gauss–Seidel method; interpret your results.
 - (b) Apply several iterations of the Krawczyk method; interpret your results.
 - (c) Apply several iterations of the interval Newton method you obtain by using the linear system solution boulder `verifylss` in INTLAB.
19. Explain why, if $F'(x)$ is a Jacobian matrix for $F : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $Y \in \mathbb{R}^{n \times n}$ is an n by n matrix, then $Y(F'(x))$ represents the Jacobian matrix for YF .
20. Prove Proposition 8.3 on page 466.
21. Fill in the details of the relationship, presented on page 467, of the interval Gauss–Seidel method to the univariate interval Newton method.
22. Fill in the details of the proof of Theorem 8.11 (on page 468).
23. By writing the quantities down in terms of sums of partial derivatives, show that $I - YF'(x)$ is the Jacobi matrix for $G(x) = x - YF(x)$, where $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$.
24. Show that if $n = 1$, the quasi-Newton equation (Equation (8.75) on page 473) reduces to Equation (8.76).
25. Prove that, for $v \in \mathbb{R}^n$ and $w \in \mathbb{R}^n$, $\|v + w\|_2 = \|v\|_2 + \|w\|_2$ if and only if $v = \alpha w$ for some positive scalar α . Use this fact to show that the strict inequality in (8.80) on page 475 holds.
26. Let F be as in Exercise 9 on page 483. Do several iterations of Broyden's method, using the same starting points as you did for Exercise 9; observe not only $x^{(k)}$, but also B_k . What do you observe? Do you observe superlinear convergence?
27. Suppose $f(x) = (x - 2)(x + 2)$, $g(x) = x^2 - 5x + 4$, and form $H(y) : \mathbb{R}^2 \rightarrow \mathbb{R}^1$ according to (8.92) (on page 480).
 - (a) Using $\epsilon = 0.4$ in (8.94) and using Newton's method on (8.95), with normalization equation (8.96), follow the curve $H(y) = 0$ from $t = 0$, $x = -2$ to $t = 1$.
 - (b) Draw a picture of the curve in $\mathbb{R}^2$, and draw your predictor steps and corrector steps on that curve.

Chapter 9

Optimization

Classical optimization involves finding the minimum or maximum of a function $\varphi : D \subset \mathbb{R}^n \rightarrow \mathbb{R}$ with respect to its argument $x \in \mathbb{R}^n$. The function φ is called the *objective function*. Sometimes, there are no restrictions on the argument x , in which case the problem is said to be *unconstrained*. Often there are side conditions, or *constraints*, expressed as inequalities and equations; the problem is then said to be *constrained*. A general optimization problem can be expressed as

$$\begin{array}{ll} \text{minimize } \varphi(x) \\ \text{subject to } c_i(x) = 0, i = 1, \dots, m_1, \\ \quad \quad \quad g_i(x) \leq 0, i = 1, \dots, m_2, \\ \text{where } \varphi : D \rightarrow \mathbb{R} \text{ and } c_i, g_i : D \rightarrow \mathbb{R}, \\ \text{and } x = (x_1, \dots, x_n)^T. \end{array} \quad (9.1)$$

(Thus, the problem is unconstrained if $m_1 = m_2 = 0$ and $D = \mathbb{R}^n$.) We will write

$$c(x) : \mathbb{R}^n \rightarrow \mathbb{R}^{m_1} = (c_1(x), \dots, c_{m_1}(x))^T$$

and

$$g(x) : \mathbb{R}^n \rightarrow \mathbb{R}^{m_2} = (g_1(x), \dots, g_{m_2}(x))^T.$$

Sometimes, some of the inequality constraints are simple, of the form $x_i \geq \underline{x}_i$, $x_i \leq \bar{x}_i$, that is, $x_i \in \mathbf{x}_i = [\underline{x}_i, \bar{x}_i]$. In such cases, the constraints are called *bound constraints*. Algorithms for solving (9.1) often gain efficiency by treating bound constraints specially.

REMARK 9.1 The optimization problem (9.1) is often called a *nonlinear program*. This term comes not from computer programming, but from the fact that optimization problems often come from operations research, where the solution to the problem provides managers with a program of production and distribution to follow. Along these lines, if φ represents a total cost, then the “objective” is to follow a program that minimizes the cost. If the functions φ , c_i , and g_i are linear, then the problem is called a *linear program*, and the process of writing down and solving such problems is called *linear programming*. $\square$

Two aspects of the solution to Problem (9.1) should be distinguished.

DEFINITION 9.1 An optimum of Problem 9.1 is a value $\bar{\varphi}$ taken on by φ at least at one point x^* for which $c(x^*) = 0$ and $g(x^*) \leq 0$, such that $\varphi(x) \geq \bar{\varphi}$ for every x satisfying $c(x) = 0$ and $g(x) \leq 0$. Every such point x^* is called an optimizer or optimizing point for Problem 9.1.

The general optimization problem is very difficult to solve; in fact, it is known to belong to a class of problems computer scientists call *NP-complete*. If a problem belongs to the NP-complete class, then no algorithm is known whose execution time is guaranteed to be $\mathcal{O}(n^k)$ for every instance of the problem (that is, for every choice of φ , c , and g). (If such a k exists, the algorithm is said to execute in *polynomial time*.) From a practical point of view, there is no general algorithm that solves (9.1) in a practical amount of time for every choice of φ , c , and g . Nonetheless, there are subclasses of problem (9.1) for which general algorithms can be designed that will always finish in a practical amount of time. Moreover, recently, sophisticated algorithms have been developed for problem (9.1) that complete on present computers in a practical amount of time, for many φ , c and g arising in applications.

An important subclass of optimization problems consists of those instances of (9.1) in which the objective φ , the equality constraint functions c , and the inequality constraint functions g are all linear functions of the parameters x ; such optimization problems are called *linear programs*. We discuss solution of linear programs in Section 9.4, starting on page 503.

Another important subclass of optimization problems, containing the class of linear programs, is when φ and the c and g are convex. (See Definition 8.7 on page 455.) Such problems, termed *convex optimization problems*, or *convex programs* can be solved in polynomial time. (Sometimes, problems that can be solved in a practical amount of time are called *tractable*.)

The set of $x \in D$ satisfying the constraints $c(x) = 0$ and $g(x) \leq 0$ is called the *feasible set*. Some practical problems have no objective function (or, equivalently, can be considered to have a constant objective function). These problems are called *constraint satisfaction problems*.

Optimization is a burgeoning field, with researchers and practitioners from many departments, such as mathematics, computer science, engineering departments, operations research and other business-oriented departments, and industrial laboratories. A thorough treatment of algorithms for all subclasses of problems and applications of practical interest is outside the scope of this text. However, we will cover certain basic principles and present the overall elements of some of the most prominent algorithms.

We consider algorithms for finding the overall minimum of φ over the entire feasible set in Section 9.6, starting on page 518. A related problem, often solved in practice because it is much easier, is the *local optimization* problem, in which we merely seek a point x such that $\varphi(y) \geq \varphi(x)$ for every y in the feasible set and in a sufficiently small ball in $\mathbb{R}^n$ about x . We discuss local optimization in the next section. Several references for the material presented in this chapter are [11], [32], [38], [54], [61], [65], [91], and [93].

9.1 Local Optimization

Local optimization corresponds to the concept of local convergence introduced in the chapter on methods for nonlinear systems of equations. In particular, we start with an initial guess, and use some iterative method to find a point x^* such that x^* satisfies the constraints $c(x^*) = 0$ and $g(x^*) \leq 0$, and $\varphi(x^*)$ is the smallest value of φ within some ball in $\mathbb{R}^n$ containing x^* . This is in contrast to *global optimization*, in which we try to find the global optimum of φ over *all* x within the domains of φ , c , and g .

9.1.1 Introduction to Unconstrained Local Optimization

In the next two sections, we seek a *local minimizer* of a function $\varphi : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^1$. That is,

$$\begin{aligned} &\text{we seek } x^* \in D \text{ such that for some } \delta > 0, \varphi(x^*) \leq \varphi(x) \text{ for all} \\ &x \in D \text{ such that } \|x - x^*\| \leq \delta, \text{ i.e., for all} \\ &x \in \overline{S}(x^*, \delta) = \{x \in D : \|x - x^*\| \leq \delta\}. \end{aligned} \quad (9.2)$$

See Figure 9.1. .

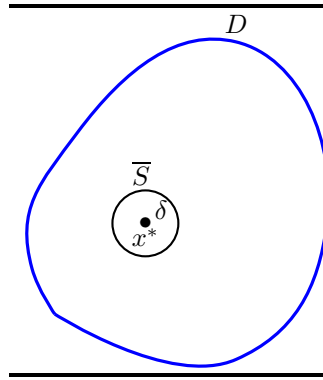


FIGURE 9.1: A local minimizer x^* only minimizes over some $\overline{S}$.

Example 9.1

$\varphi(x) = \varphi(x_1, x_2) = x_1^2 + x_2^2$ and $D = \{(x_1, x_2)^T \in \mathbb{R}^2 : -1 \leq x_1, x_2 \leq 1\}$. For this example, $x^* = (0, 0)^T$ is a local minimizer. In this example, x^* also happens to be a *global minimizer* of φ , that is $\varphi(x^*) \leq \varphi(x)$ for all $x \in D$. $\square$

Iterative methods for finding an approximate local optimum of an objective function φ of n variables often repeatedly find an optimum of a function of one variable. Such univariate optimization, used in this context, is often called a *line search*. We consider a particular line search method in the next section.

9.1.2 Golden Section Search

Consider the problem of finding the minimum of scalar function $\varphi(x)$ on the interval $[a, b]$. Assume that $\varphi(x)$ has a unique minimum at the point $x^* \in (a, b)$. The golden section search procedure to find x^* , consists of the following steps:

1. Set

$$\begin{aligned}x_0 &\leftarrow a, \\x_1 &\leftarrow a + (b - a)(1 - \alpha), \\x_2 &\leftarrow a + (b - a)\alpha, \\x_3 &\leftarrow b,\end{aligned}$$

for some α , $1/2 < \alpha < 1$. (An optimal α is $\alpha = (-1 + \sqrt{5})/2$, as we will show later.)

2. *DO WHILE* ($|\hat{x}_3 - \hat{x}_0| > \epsilon$)

(a) *IF* $\varphi(x_1) > \varphi(x_2)$, *THEN* $x^* \in (x_1, x_3)$, and

- (i) $\hat{x}_0 \leftarrow x_1$ and $\hat{x}_3 \leftarrow x_3$;
- (ii) $\hat{x}_2 \leftarrow \hat{x}_0 + (\hat{x}_3 - \hat{x}_0)\alpha$;
- (iii) $\hat{x}_1 \leftarrow \hat{x}_0 + (\hat{x}_3 - \hat{x}_0)(1 - \alpha)$.

ELSE $x^* \in (x_0, x_2)$, and

- (i) $\hat{x}_0 \leftarrow x_0$ and $\hat{x}_3 \leftarrow x_2$;
- (ii) $\hat{x}_2 \leftarrow \hat{x}_0 + (\hat{x}_3 - \hat{x}_0)\alpha$;
- (iii) $\hat{x}_1 \leftarrow \hat{x}_0 + (\hat{x}_3 - \hat{x}_0)(1 - \alpha)$.

END IF

(b) *IF* $|\hat{x}_3 - \hat{x}_0| \leq \epsilon$ *THEN*

OUTPUT: $(\hat{x}_0 + \hat{x}_3)/2$

ELSE

$$x_0 \leftarrow \hat{x}_0, x_1 \leftarrow \hat{x}_1, x_2 \leftarrow \hat{x}_2, x_3 \leftarrow \hat{x}_3.$$

END IF

END DO

REMARK 9.2 In Step 2(a) of the above procedure,

$$\frac{\hat{x}_1 - \hat{x}_0}{\hat{x}_3 - \hat{x}_0} = 1 - \alpha \quad \text{and} \quad \frac{\hat{x}_2 - \hat{x}_0}{\hat{x}_3 - \hat{x}_0} = \alpha$$

for each iteration. In fact, we can find an α so that, simultaneously, $\hat{x}_1 \leftarrow \hat{x}_0 + (\hat{x}_3 - \hat{x}_0)(1 - \alpha)$ in the third branch of step 2a can be replaced by $\hat{x}_1 \leftarrow x_2$ and $\hat{x}_2 \leftarrow \hat{x}_0 + (\hat{x}_3 - \hat{x}_0)\alpha$ in the second branch of step 2a can be replaced by $\hat{x}_2 \leftarrow x_1$. For this simplification, α must satisfy

$$\begin{aligned} x_2 &= x_1 + (x_3 - x_1)(1 - \alpha) \quad \text{and} \\ x_1 &= x_0 + (x_2 - x_0)\alpha. \end{aligned}$$

Solving these equations for α (utilizing the relationships between x_0 , x_1 , x_2 , and x_3 defined in step 1) gives

$$\alpha = \frac{-1 + \sqrt{5}}{2} \approx .618.$$

(The number $1 + \alpha$, ubiquitous in classical elementary geometry, is sometimes called the *golden mean*.) Using this α , we only need to define one new point on each iteration. The significance of this is that only one evaluation of φ is required per iteration.¹ $\square$

REMARK 9.3 If $[x_0^{(k)}, x_3^{(k)}]$ is the k -th interval in the golden section procedure with $x_0^{(0)} = a$ and $x_3^{(0)} = b$, it is easy to show that $x_3^{(k)} - x_0^{(k)} = \alpha^k(b - a)$, where $\alpha = (-1 + \sqrt{5})/2$. Thus, since $x^* \in [x_0^{(k)}, x_3^{(k)}]$, we have

$$\left| x^* - \frac{x_0^{(k)} + x_3^{(k)}}{2} \right| \leq \frac{1}{2} \alpha^k (b - a), \quad \text{for } k = 0, 1, 2, \dots$$

$\square$

9.1.3 Relationship to Nonlinear Systems

We now return to consideration of problem (9.2) in the general n -dimensional setting with $n \geq 1$. In the remainder of this section, we assume φ is differentiable. In this case, trying to find a zero of $\nabla\varphi$, the gradient of φ , is usually part of the process for solving (9.2). This approach is based on the fact that if x^* is a local minimizer of φ on an open set D and φ is differentiable at x^* , then necessarily $\nabla\varphi(x^*) = 0$. Since $\nabla\varphi(x) = 0$ consists of n equations in the unknown components of x , we see that the minimization problem, when

¹In general, φ may be quite complicated, so reducing the number of evaluations of φ significantly improves the algorithm's efficiency.

φ is differentiable, leads to finding the solutions to a system of n nonlinear equations in n unknowns. (Points x at which $\nabla\varphi(x) = 0$ are called *critical points* of the unconstrained optimization problem.)

Example 9.2

Consider $\varphi(x) = x_1^2 + x_2^2$. Then

$$\nabla\varphi(x) = \left(\frac{d\varphi}{dx_1}, \frac{d\varphi}{dx_2} \right)^T = (2x_1, 2x_2)^T.$$

Thus, $\nabla\varphi(x) = 0$ when $x_1 = x_2 = 0$. □

We may therefore attempt application of the methods of the previous chapter to finding the solution of $\nabla\varphi(x) = 0$. For instance, since $F = \nabla\varphi(x)$ is a mapping from $\mathbb{R}^n$ to $\mathbb{R}^n$, we may use (when φ is twice differentiable) Newton's method, which here takes the form

$$x^{(k+1)} = x^{(k)} - (\nabla^2\varphi(x^{(k)}))^{-1}\nabla\varphi(x^{(k)}) \quad \text{for } k = 0, 1, 2, \dots \quad (9.3)$$

where $\nabla^2\varphi(x)$ is *Hessian matrix* of φ at x , i.e., $\nabla^2\varphi(x)$ is the Jacobian matrix of $\nabla\varphi(x)$. Under appropriate conditions, we can obtain local and quadratic convergence of (9.3) to a zero of $\nabla\varphi$.

REMARK 9.4 The Hessian matrix has the form

$$\nabla^2\varphi(x) = \begin{pmatrix} \frac{\partial^2\varphi(x)}{\partial x_1^2} & \frac{\partial^2\varphi(x)}{\partial x_2\partial x_1} & \cdots & \frac{\partial^2\varphi(x)}{\partial x_n\partial x_1} \\ \frac{\partial^2\varphi(x)}{\partial x_1\partial x_2} & \frac{\partial^2\varphi(x)}{\partial x_2^2} & \cdots & \frac{\partial^2\varphi(x)}{\partial x_n\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2\varphi(x)}{\partial x_1\partial x_n} & \frac{\partial^2\varphi(x)}{\partial x_2\partial x_n} & \cdots & \frac{\partial^2\varphi(x)}{\partial x_n^2} \end{pmatrix}.$$

□

REMARK 9.5 Even though $\nabla\varphi(x) = 0$ when x is a local minimizer of φ , there are many places other than at local minimizers where $\nabla\varphi$ can equal 0. Thus, practical algorithms for minimization of φ may need to include procedures in addition to solving $\nabla\varphi = 0$. □

REMARK 9.6 We can also transform the problem of finding the solution to any nonlinear problem into a minimization problem. Suppose that we seek

x such that $F(x) = (f_1(x), f_2(x), \dots, f_n(x))^T = 0$. Define

$$\varphi(x) = \sum_{i=1}^n (f_i(x))^2.$$

Then F will have a zero when the function φ has a global minimum, although there may be local minima of φ that do not correspond to zeros of F , and φ may have a minimum, even though F has no zeros. (Notice that $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ while $\varphi : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^1$.) $\square$

We now briefly consider three other methods for numerically solving the minimization problem (9.2). First, we consider the method of steepest descent.

9.1.4 Steepest Descent

Recall that we seek $\varphi(x^*)$ such that $\varphi(x^*) \leq \varphi(x)$ for all $x \in D$ such that $\|x - x^*\| \leq \delta$.

In descent methods for finding x^* , we pick $x^{(0)}$, then try to find direction $s^{(0)}$ such that $x^{(1)} = x^{(0)} + \lambda_k s^{(0)}$ satisfies $\varphi(x^{(1)}) < \varphi(x^{(0)})$ for sufficiently small λ_k . In general, a descent method generates a direction $s^{(k)}$ of local descent for each iterate $x^{(k)}$, in the sense that there is a λ_k^* such that $\varphi(x^{(k)} + \lambda s^{(k)}) < \varphi(x^{(k)})$ for $\lambda \in (0, \lambda_k^*]$. The next iterate is of the form

$$x^{(k+1)} = x^{(k)} + \lambda_k s^{(k)}, \quad (9.4)$$

where λ_k is chosen (according to one of various strategies) to assure $\varphi(x^{(k+1)}) < \varphi(x^{(k)})$. The direction $s^{(k)}$ and the parameters λ_k should be chosen so the sequence $\{\nabla\varphi(x^{(k)})\}$ converges to 0. If $\|\nabla\varphi(x^{(k)})\|$ is small, then usually $x^{(k)}$ is near a zero of $\nabla\varphi$, while the fact that the sequence $\{\varphi(x^{(k)})\}$ is decreasing indicates that this zero of $\nabla\varphi$ is probably a local minimizer of φ .

The simplest example is the *method of steepest descent*, for which we ask for a vector $\hat{s}$ of unit length with respect to the L_2 -norm (i.e., $\|\hat{s}\|_2 = 1$) such that the directional derivative $D_{\hat{s}}\varphi(x)$ is minimum over all directional derivatives $D_s\varphi(x)$. Assuming that $\nabla\varphi(x) \neq 0$, $\hat{s} = -\nabla\varphi(x)/\|\nabla\varphi(x)\|_2$, since the directional derivative of φ is minimized in direction $-\nabla\varphi(x)$. Thus, the method of steepest descent is given by

$$x^{(k+1)} = x^{(k)} - \lambda_k \nabla\varphi(x^{(k)}), \quad k = 0, 1, 2, \dots, \quad (9.5)$$

where λ_k is chosen to guarantee that $\varphi(x^{(k+1)}) < \varphi(x^{(k)})$.

The following result guarantees the existence of such a parameter.

LEMMA 9.1

Let $\varphi : \mathbb{R}^n \rightarrow \mathbb{R}$ be defined on an open set D and differentiable at x in D . If $[\nabla\varphi(x)]^T s < 0$ for some $s \in \mathbb{R}^n$, then there is a $\lambda^* = \lambda^*(x, s)$ such that $\lambda^* > 0$ and $\varphi(x + \lambda s) < \varphi(x)$ for all $\lambda \in (0, \lambda^*)$.

PROOF The proof follows from the fact that

$$\lim_{\lambda \rightarrow 0^+} \frac{\varphi(x + \lambda s) - \varphi(x)}{\lambda} = [\nabla \varphi(x)]^T s. \quad (9.6)$$

□

This lemma guarantees, in particular, that the parameter λ_k in the steepest descent method can be chosen such that $\varphi(x^{(k+1)}) < \varphi(x^{(k)})$. This is not sufficient to show that $\{x^{(k)}\}$ approaches a zero of $\nabla \varphi$, since λ_k may be arbitrarily small. (For example, λ_k might happen to be chosen so that $\|x^{(k+1)} - x^{(k)}\| < \epsilon/2^k$, with the result that $\{x^{(k)}\}$ converges to a point $\tilde{x}$ such that $\|x^{(0)} - \tilde{x}\| < 2\epsilon$.)

We now turn to a selection of λ_k in descent methods of the form

$$x^{(k+1)} = x^{(k)} + \lambda_k s^{(k)}, \quad (9.7)$$

where $s^{(k)} = -\nabla \varphi(x^{(k)})$ for the steepest descent method. Consider

$$x^{(k+1)} = x^{(k)} - \lambda_k \nabla \varphi(x_k).$$

We wish to find λ that minimizes the scalar function

$$h(\lambda) = \varphi(x^{(k+1)}) = \varphi(x^{(k)} - \lambda \nabla \varphi(x^{(k)}))$$

for λ in an interval $[0, \lambda^*]$. One way would be to use the golden section search procedure, assuming λ^* is large enough to guarantee that a minimum of h is in the interval $[0, \lambda^*]$. Another way would involve differentiating h and determining the critical points of h directly; this is generally too costly a procedure. A third approach begins with selecting three nonnegative estimates λ_{k_1} , λ_{k_2} , λ_{k_3} to λ_k . Then, the quadratic polynomial interpolant to h through λ_{k_1} , λ_{k_2} , and λ_{k_3} is calculated. Next, λ_k is defined to be the number that minimizes this quadratic polynomial. For example, if λ_{k_1} , λ_{k_2} , and λ_{k_3} are set equal to 0, $\frac{1}{2}$, and 1, respectively, then

$$p(\lambda) = g_1 + h_1 \lambda + h_3 \lambda \left(\lambda - \frac{1}{2} \right)$$

interpolates $h(\lambda)$ at $\lambda = 0$, $\frac{1}{2}$, and 1, where

$$\begin{aligned} g_1 &= \varphi(x^{(k)}), \\ g_2 &= \varphi \left(x^{(k)} - \frac{1}{2} \nabla \varphi(x^{(k)}) \right), \\ g_3 &= \varphi(x^{(k)} - \nabla \varphi(x^{(k)})), \\ h_1 &= 2(g_2 - g_1), \\ h_2 &= 2(g_3 - g_2), \text{ and} \\ h_3 &= h_2 - h_1. \end{aligned}$$

Then, let

$$g_\alpha = \varphi(x^{(k)} - \alpha \nabla \varphi(x^{(k)})),$$

where $\alpha = 1/4 - h_1/(2h_3)$ is the critical point of the polynomial $p(\lambda)$. Next, λ_k is selected from $\{\alpha, 0, \frac{1}{2}, 1\}$ such that

$$\varphi(x^{(k)} - \lambda_k \nabla \varphi(x^{(k)})) = \min\{g_\alpha, g_1, g_2, g_3\}.$$

Finally, $x^{(k+1)} = x^{(k)} - \lambda_k \nabla \varphi(x^{(k)})$.

REMARK 9.7 There are many variations in the method of steepest descent. In particular, intricate methods have been devised for determining λ_k . However, in general, steepest descent methods are only linearly convergent, but will converge independently of the starting approximation. Of course, the methods may converge to a minimum that is not the absolute minimum of φ . $\square$

REMARK 9.8 Unlike Newton's method for nonlinear systems of equations, the steepest descent method for nonlinear optimization is sensitive to scaling, and is quite sensitive to the condition number of the problem. In particular, convergence can be very slow if the ratio of the largest eigenvalue to the smallest eigenvalue² of the Hessian matrix is large. $\square$

9.1.5 Quasi-Newton Methods for Minimization

In quasi-Newton methods, the emphasis is on finding particular solutions to the system of nonlinear equations $\nabla \varphi = 0$ that correspond to local minima of φ . When we speak of *quasi*-Newton methods, we usually mean that we use an iteration matrix constructed with the quasi-Newton equation (Equation 8.75 on page 473), rather than the Jacobian matrix of F . For the minimization problem $F = \nabla \varphi$, and the Jacobian matrix is the Hessian matrix of φ . We call the formula for obtaining a new approximation to the Jacobian matrix according to (8.75) an *update*. Special quasi-Newton updates (other than Broyden's method, explained in §8.5) can be designed to appropriately approximate Hessian matrices near local minimizers. We now describe one of these.

9.1.5.1 The Davidon–Fletcher–Powell and BFGS Updates

Recall that we seek $x^* \in D$ such that for some $\delta > 0$, $\varphi(x^*) \leq \varphi(x)$ for all $x \in D$ such that $\|x - x^*\| \leq \delta$ where $\varphi : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^1$. If we use Newton's method to solve this problem, we seek a solution x^* of $\nabla \varphi(x^{(k)}) = 0$, and

²each of which must be positive at a local minimum

Newton's method has the form

$$x^{(k+1)} = x^{(k)} - (\nabla^2 \varphi(x^{(k)}))^{-1} \nabla \varphi(x^{(k)}) \quad (9.8)$$

where $\nabla^2 \varphi(x^{(k)})$ is the Hessian matrix. (Note that $\nabla^2 \varphi(x^{(k)})$ is symmetric and positive definite at $x = x^*$.)

In quasi-Newton methods, we replace $\nabla^2 \varphi(x^{(k)})$ by some approximation B_k , thus obtaining

$$x^{(k+1)} = x^{(k)} - B_k^{-1} \nabla \varphi(x^{(k)}). \quad (9.9)$$

One choice of quasi-Newton method for solving (9.9) would be Broyden's method. However, since $\nabla^2 \varphi(x^{(k)})$ is symmetric and positive definite (for $x^{(k)}$ near x^*), it would be desirable to have these features in B_k .

The Davidon–Fletcher–Powell (DFP quasi-Newton update provides these features, i.e., if B_k is symmetric positive definite, then B_{k+1} is symmetric positive definite. This method has the form:

$$x^{(k+1)} = x^{(k)} - B_k^{-1} \nabla \varphi(x^{(k)}). \quad (9.10)$$

$$y = \nabla \varphi(x^{(k+1)}) - \nabla \varphi(x^{(k)}), \quad s = x^{(k+1)} - x^{(k)} \quad (9.11)$$

$$B_{k+1} = (I - \frac{ys^T}{y^T s}) B_k (I - \frac{sy^T}{y^T s}) + \frac{yy^T}{y^T s}. \quad (9.12)$$

It can be shown, using similar techniques to those in the analysis of Broyden's method, that, if

$$H_{k+1} = H_k + \frac{ss^T}{s^T y} - \frac{H_k y y^T H_k}{y^T H_k y}$$

and $H_k = B_k^{-1}$, then $H_{k+1} = B_{k+1}^{-1}$. Thus, the Davidon–Fletcher–Powell method can be implemented as

$$x^{(k+1)} = x^{(k)} - H_k \nabla \varphi(x^{(k)}). \quad (9.13)$$

$$y = \nabla \varphi(x^{(k+1)}) - \nabla \varphi(x^{(k)}), \quad s = x^{(k+1)} - x^{(k)} \quad (9.14)$$

$$H_{k+1} = H_k + \frac{ss^T}{s^T y} - \frac{H_k y y^T H_k}{y^T H_k y}. \quad (9.15)$$

Here, $x^{(0)}$ is an initial guess, and H_0 is either $(\nabla^2 \varphi(x^{(0)}))^{-1}$ or a good approximation to $(\nabla^2 \varphi(x^{(0)}))^{-1}$.

There are also other, perhaps better, quasi-Newton updates for solving minimization problems, such as the Broyden–Fletcher–Goldfarb–Shanno (BFGS) update; see [25, p. 457]. The BFGS update is defined by

$$\tilde{H}_{k+1} = \tilde{H}_k + \frac{(s - \tilde{H}_k)s^T + s(s - \tilde{H}_k y)}{s^T y} - \frac{y^T (s - \tilde{H}_k y) s s^T}{(s^T s)^2}, \quad (9.16)$$

where $\tilde{H}_k$ is the k -th approximate inverse for the BFGS update, and where y and s are as in (9.14).

A classic reference for methods that incorporate quasi-Newton methods is [24].

9.1.6 The Nelder–Mead Simplex (Direct Search) Method

In this section, we consider the downhill simplex method of Nelder and Mead for finding a minimum of $\varphi(x)$ where $\varphi : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^1$. The method requires only function evaluations. This heuristic³ search method is practical in many cases for a small number of variables, and is versatile, since the only information about φ that is required is, if $x_1 \in \mathbb{R}^n$ and $x_2 \in \mathbb{R}^n$ are in the domain of φ , we can determine⁴ whether or not $\varphi(x_1) \geq \varphi(x_2)$. Virginia Torczon has analyzed convergence of this method and generalizations of it, termed *pattern search algorithms*. The main disadvantages of the method are slow convergence when high accuracy is required and impractically slow performance on high-dimensional problems.

REMARK 9.9 The Nelder-Mead simplex method is used in the MATLAB function `fminsearch`. $\square$

ALGORITHM 9.1

(The simplex method of Nelder and Mead)

INPUT:

1. the initial point $x^{**} \in \mathbb{R}^n$;
2. the heuristic parameters λ , α , β , and γ . (Here, λ is related to the problem's length scale.)
3. the stopping tolerance ϵ .

OUTPUT: an approximate optimizer x and an approximate optimum $\varphi(x)$.

1. (Assign the initial simplex; see Remark 9.10 following this algorithm.)

$$(a) \ x_1 \leftarrow x^{**}.$$

$$(b) \ x_{i+1} \leftarrow x_1 + \lambda e_i \text{ for } i = 1, 2, 3, \dots, n.$$

2. Order the points $x_1, x_2, \dots, x_{n+1}$ so that

$$\varphi_{n+1} \geq \varphi_n \geq \varphi_{n-1} \geq \dots \geq \varphi_2 \geq \varphi_1,$$

where $\varphi_k = \varphi(x_k)$.

³A heuristic is a rule of thumb that is used to determine whether or not a mathematical property is true. If the property is true according to the heuristic, then in actuality, the property will be true in many cases, but not in all cases. Thus, a heuristic method may not always find the answer, but often does.

⁴This determination can be done, say, interactively. For example, if φ corresponds to a polynomial fit to data points, the user can be presented with a graph of the polynomial corresponding to x_1 and a graph corresponding to x_2 , then the user can tell the computer which one is better.

3. (Construct a new point. At each iteration, a new simplex is produced by replacing the “worst” point x_{n+1} , the point with the highest value of φ .) Let

$$x_c \leftarrow \frac{1}{n} \sum_{j=1}^n x_j$$

be the centroid of the n best vertices. In this step, we construct

$$x_r \leftarrow x_c + \alpha(x_c - x_{n+1}),$$

where x_{n+1} is the point with the highest function value and where α is a reflection coefficient.

Set $\varphi_r \leftarrow \varphi(x_r)$.

Case 1: IF $\varphi_1 \leq \varphi_r \leq \varphi_n$ (x_r is not a new best point or a new worst point), THEN $x_{n+1}^* \leftarrow x_r$.

Case 2: IF $\varphi_r < \varphi_1$ (x_r is the new best point), then we assume that the direction of reflection is good direction, and we attempt to expand the simplex in this direction. We define an expanded point by

$$x_e \leftarrow x_c + \beta(x_r - x_c),$$

where $\beta > 1$ is an expansion coefficient.

IF $\varphi_e < \varphi_r$

THEN the expansion is successful, and $x_{n+1}^* \leftarrow x_e$.

OTHERWISE, the expansion failed, and $x_{n+1}^* \leftarrow x_r$.

Case 3: IF $\varphi_r > \varphi_n$, the simplex is assumed to be too large, and should be contracted. A contraction step is carried out, where

$$x_c \leftarrow \begin{cases} x_c + \gamma(x_{n+1} - x_c) & \text{if } \varphi_r \geq \varphi_{n+1}, \\ x_c + \gamma(x_r - x_c) & \text{if } \varphi_r \leq \varphi_{n+1}, \end{cases}$$

where $0 \leq \gamma \leq 1$ is a contraction coefficient.

IF $\varphi_c \leq \min(\varphi_r, \varphi_{n+1})$, the contraction step has succeeded, and

$$x_{n+1}^* = x_c.$$

OTHERWISE, a further contraction is carried out. (That is, repeat this step.)

4. IF

$$\sqrt{\sum_{j=1}^{n+1} (\varphi_j - \mu)^2 / n} < \epsilon, \quad \text{where } \mu = \sum_{j=1}^{n+1} \varphi_j / (n+1),$$

THEN continue to step 5.

OTHERWISE:

(a) $x_{n+1} \leftarrow x_{n+1}^*$;

(b) Return to step 2.

5. (If the standard deviation of the function values is smaller than a specific tolerance ϵ , then the search terminates for the starting guess x^{**} .)

IF $\|x^{**} - x_1\| > \epsilon$, THEN

(a) $x^{**} \leftarrow x_1$;

(b) restart the problem with one vertex of the simplex at the new minimum point x_1 . That is, return to step 1 after setting $x^{**} \leftarrow x_1$.

(Restarting is performed so that the criterion in step 4 is not fooled by a single anomalous step that, for some reason, failed to move the worst point by more than ϵ .)

(c) IF however, $\|x^{**} - x_1\| < \epsilon$, THEN

i. OUTPUT: x_1 and $\varphi(x_1)$.

ii. HALT.

END ALGORITHM 9.1.

REMARK 9.10 The $n + 1$ points $x_1, x_2, \dots, x_{n+1}$ in Step 1 of Algorithm 9.1 define an n -dimensional simplex. A simplex is a geometrical figure consisting in n dimensions of $n + 1$ vertices, all interconnecting line segments, polygonal faces, etc.; see Figure 9.2. $\square$

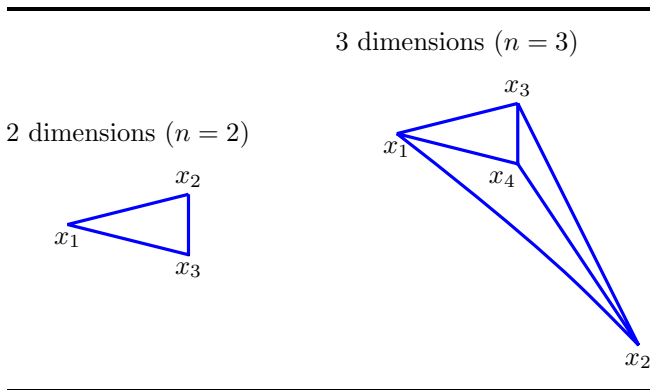


FIGURE 9.2: Illustration of 2- and 3-dimensional simplexes.

Example 9.3

Consider $\varphi(x) = x^2 + y^2 - 4$.

Initialization: Set $x^{**} \leftarrow (-2, 3)$, $\lambda \leftarrow 1$, $\alpha \leftarrow 1$, $\beta \leftarrow 2$, $\gamma \leftarrow 1/2$, and $\epsilon \leftarrow .01$.

Step 1: $x_2 \leftarrow (-1, 3)$, $x_3 \leftarrow (-2, 4)$, $x_1 = x^{**} = (-2, 3)$.

Step 2: $\varphi(x_1) = 9$, $\varphi(x_2) = 6$, $\varphi(x_3) = 16$. Thus, set $x_1 \leftarrow (-1, 3)$, $x_2 \leftarrow (-2, 3)$, and $x_3 \leftarrow (-2, 4)$ (reorder the points).

Step 3:

$$x_c \leftarrow \frac{1}{2} \sum_{i=1}^2 x_i = \left(-\frac{3}{2}, 3\right),$$

$$x_r \leftarrow x + \alpha(x_c - x_{n+1}) = (-1, 2), \quad \varphi_r \leftarrow \varphi(x_r) = 1.$$

Case 2: Perform an expansion, since $\varphi_r < \varphi_1$:

$$x_e \leftarrow x_c + \beta(x_r - x_c) = 2x_r - x_c = \left(\frac{1}{2}, -1\right).$$

Since $\varphi(x_e) = -\frac{11}{4} < \varphi(x_r)$, the expansion is successful, so set $x_3^* \leftarrow x_e = \left(\frac{1}{2}, -1\right)$.

Step 4: $\|x_3^* - x_3\|_\infty = 5 > \epsilon$. Set $x_3 \leftarrow x_3^* = \left(\frac{1}{2}, -1\right)$, then return to step 2.

Step 2 $\varphi(x_1) = 6$, $\varphi(x_2) = 9$, and $\varphi(x_3) = -\frac{11}{4}$. Thus, reorder:

$$x_1 = \left(\frac{1}{2}, -1\right), \quad x_2 = (-1, 3), \quad x_3 = (-2, 3).$$

(This results in $\varphi_1 = -\frac{11}{4}$, $\varphi_2 = 6$, and $\varphi_3 = 9$.)

Step 3:

$$x_c \leftarrow \frac{1}{2} \sum_{i=1}^2 x_i = \left(-\frac{1}{4}, 1\right),$$

$$x_r \leftarrow x_c + \alpha(x_c - x_3) = \left(\frac{3}{2}, -1\right), \quad \varphi_r \leftarrow \varphi(x_r) = -\frac{3}{4}.$$

Case 1: $\varphi_1 \leq \varphi_r \leq \varphi_2$. Thus, set

$$x_3^* \leftarrow x_r = \left(\frac{3}{2}, -1\right).$$

Step 4 $\|x_3^* - x_3\|_\infty = 4 > \epsilon$. Thus, set $x_3 \leftarrow x_3^* = \left(\frac{3}{2}, -1\right)$, then return to step 2 and continue with the algorithm.

□

9.1.7 Software for Unconstrained Local Optimization

General software for unconstrained local optimization typically uses combinations of steepest descent and Newton or quasi-Newton methods in sophisticated ways. Some local optimization algorithms are embedded into interactive systems such as MATLAB, Mathematica, or Maple. When using such systems for a particular problem, one should be cautious about believing that the calculated result is indeed the global optimum (to within roundoff error), even if the routine exits with no reported error. There is often no guarantee that, if the algorithm returns a point x without reporting an error, then x is a local optimum (unless the problem is well-understood, such as when the problem is convex). However, present-day algorithms often return reasonable results for many problems of practical interest.

Source code for unconstrained local optimization, that can be embedded in user-written software, is available from the Netlib repository, at

<http://www.netlib.org/>

as well as in various commercial and proprietary packages.

9.2 Constrained Local Optimization

Traditionally, many of the techniques used in unconstrained optimization, such as line searches, descent methods, and quasi-Newton updates, can be used for constrained optimization, but with various complications. A treatment of classic techniques for constrained local optimization appears in [32]. Much of the best software for constrained local optimization is proprietary (such as `fmincon` from the MATLAB optimization toolbox for the general constrained problem).

9.3 Constrained Optimization and Nonlinear Systems

In unconstrained optimization, the problem of minimizing φ occurs at a critical point, that is, at a point where $\nabla\varphi = 0$. An analogous set of equations for the general constrained optimization problem is derivable from the Lagrange multiplier conditions, and are usually called the *Kuhn–Tucker equations*. The Kuhn–Tucker equations for the general optimization problem (9.1)

are:

$$F(X) = \begin{pmatrix} \nabla\varphi(x) + u^T\nabla g(x) + v^T\nabla c(x) \\ u_1g_1(x) \\ \vdots \\ u_{m_2}g_{m_2}(x) \\ c_1(x) \\ \vdots \\ c_{m_1}(x) \end{pmatrix} = 0, \quad (9.17)$$

where $u \in \mathbb{R}^{m_2}$, $v \in \mathbb{R}^{m_1}$, ∇g is the matrix whose i -th column is ∇g_i , ∇c is the matrix whose i -th column is ∇c_i , and the condition $u \geq 0$ (where the inequality is interpreted componentwise) must be satisfied. Points satisfying the Kuhn–Tucker conditions (9.17) are called *critical points*, or *Kuhn–Tucker points* of the constrained optimization problem (9.1). This is a system of $n + m_1 + m_2$ equations in the unknown vectors x , u , and v . We have

THEOREM 9.1

If the functions φ , c , and g are sufficiently smooth and x^ is a local solution to the constrained optimization problem (9.1), then x^* must satisfy the Kuhn–Tucker conditions, for some admissible choice of u and v .*

One place where derivation of the Kuhn–Tucker conditions and a proof of Theorem 9.1 can be found is [32].

REMARK 9.11 The Lagrange multipliers u and v are sometimes termed *dual variables*, while the original variables x are called the *primal variables*. The values of these dual variables have practical interpretations in the real-world problems that give rise to the optimization problem. This is especially true in linear programming: In linear programming, *dual problems* can be easily formulated. In such cases, the original problem is called the *primal problem*. The dual variables of the primal problem are the primal variables of the dual problem, and the primal variables of the original problem are the dual variables of the dual problem. Many optimization algorithms rely on the interplay between solutions of the primal and the dual. A good discussion of duality in linear programs appears in [29, Chapter 4], while duality is framed in terms of linear algebra in [32]. Practical rules for forming dual problems appear in [12]. $\square$

In general, except for $u_i \geq 0$, bounds on the Lagrange multipliers u and v in the Kuhn–Tucker conditions are not known, while it is advantageous in

some algorithms to know such bounds. For this and other reasons dealing with special cases, the Kuhn–Tucker conditions are sometimes replaced by addition of an additional parameter u_0 , replacing $\nabla\varphi$ in the first equation by $u_0\nabla\varphi$, and adding an additional *normalization condition*, such as

$$\left(u_0 + \sum_{j=1}^{m_2} u_j + \sum_{i=1}^{m_1} v_i^2\right) - 1 = 0.$$

These new equations, where $u_i \in [0, 1]$, $0 \leq i \leq m_2$, and $v_i \in [-1, 1]$, are termed the *Fritz John conditions*.

9.4 Linear Programming

Linear programs are a special type of constrained optimization problem in which finding the global optimum is tractable. Before considering the general linear programming problem, a common optimization problem in business and industry, we consider a simple example problem.

Example 9.4

Find $x_1, x_2 \geq 0$ such that

$$\begin{cases} 4x_1 + 2x_2 \leq 8 \\ 2x_1 + 4x_2 \leq 8 \end{cases}$$

and such that

$$z = 3x_1 + 2x_2$$

is a maximum.

The solution of this problem is easy to find using a geometric argument. (See Figure 9.3.) Notice that the solution space is bounded by the lines $x_1 = 0$, $x_2 = 0$, $4x_1 + 2x_2 = 8$, and $2x_1 + 4x_2 = 8$. Consider $z = 3x_1 + 2x_2$, which corresponds to a family of straight lines as z varies. Notice that the value z increases as the line is moved farther from the origin. Thus, z is maximized at $(\frac{4}{3}, \frac{4}{3})$, with corresponding maximum value $z = \frac{20}{3}$. $\square$

REMARK 9.12 This graphical approach is practical only for simple example problems. $\square$

We now define the general linear programming problem.

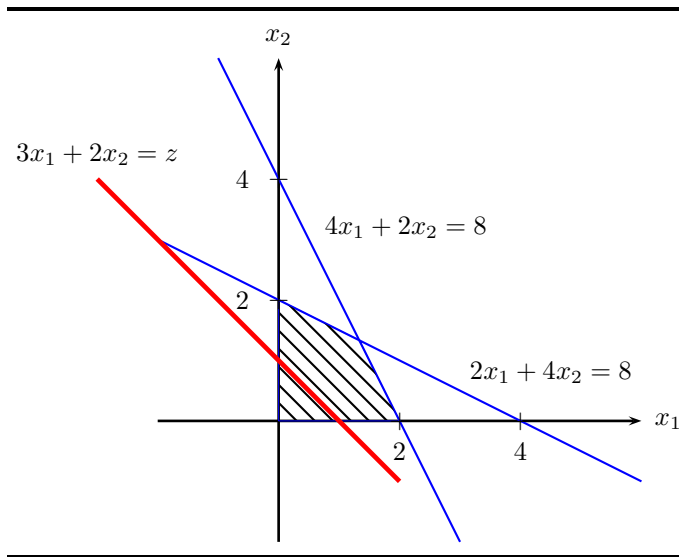


FIGURE 9.3: Graph of the constraint set and objective function z , Example 9.4.

DEFINITION 9.2 The general linear programming problem in standard form is:

maximize

$$z = c_0 + \sum_{j=1}^n c_j x_j \quad (\text{objective function})$$

subject to the linear equalities

$$\sum_{j=1}^n a_{ij} x_j = b_i \quad \text{for } i = 1, 2, \dots, m_1$$

and nonnegativity constraints

$$x_j \geq 0 \quad \text{for } j = 1, 2, \dots, n.$$

(9.18)

Assumption: $b_i \geq 0$ for each i and $n \geq m_1$ (If $n = m_1$, the values of x_i are uniquely determined when the matrix $A = \{a_{ij}\}$ is of full rank.)

We first consider how problems can be converted to the above standard form.

REMARK 9.13 The constant c_0 does not effect the optimizing point x , but only affects the optimum value z . If we ignore c_0 , then the standard form

for the general linear programming problem can be written in matrix form as

$$\begin{aligned} & \text{maximize } c^T x \\ & \text{subject to } Ax = b \\ & \text{and } x \geq 0, \end{aligned} \tag{9.19}$$

where⁵ $x \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m_1 \times n}$, $b \in \mathbb{R}^{m_1}$, and “ $\geq$ ” is interpreted componentwise. $\square$

Different texts and software packages may define “standard form” somewhat differently than above. However, one “standard form” can be converted to another, as we now illustrate.

9.4.1 Converting to Standard Form

Case A: (Converting a minimization problem to a maximization problem)

Suppose that we want to minimize

$$c_0 + \sum_{j=1}^n c_j x_j.$$

Then, set $z = -c_0 - \sum_{j=1}^n c_j x_j$, and maximize z .

Case B: (Some $b_i < 0$) If $b_i < 0$, then replace the constraint

$$\sum_{j=1}^n a_{ij} x_j = b_i$$

by

$$\sum_{j=1}^n -a_{ij} x_j = -b_i = \hat{b}_i,$$

where $\hat{b}_i = -b_i$.

Case C: (Replacing inequalities by equalities) Suppose that a constraint is

$$\sum_{j=1}^n a_{ij} x_j \leq b_i.$$

Then, we introduce a new “slack” variable $s_i \geq 0$ so that

$$\sum_{j=1}^n a_{ij} x_j + s_i = b_i.$$

⁵The vector c here has a different meaning from the vector function c used in the general form (9.1) on page 487.

Case D: (Some variables are not constrained to be nonnegative) Suppose that x_j is not constrained to be nonnegative. We introduce $x_j^+ \geq 0$ and $x_j^- \geq 0$ such that $x_j = x_j^+ - x_j^-$.

Example 9.5

Minimize $-x_1 + x_2 - x_3$ subject to

$$\begin{cases} x_1 - 3x_2 + 4x_3 = 5, \\ x_1 - 2x_2 \leq 3, \\ 2x_2 - x_3 \geq 4, \\ x_1 \geq 0, \\ x_2 \geq 0. \end{cases}$$

We convert this to standard form:

Maximize $z = x_1 - x_2 + x_3^+ - x_3^-$ subject to

$$\begin{cases} x_1 - 3x_2 + 4x_3^+ - 4x_3^- = 5, \\ x_1 - 2x_2 + x_4 = 3, \\ 2x_2 - x_3^+ + x_3^- - x_5 = 4, \end{cases}$$

where $x_1, x_2, x_3^+, x_3^-, x_4$, and x_5 are all nonnegative. Note that $n = 6$ and $m_1 = 3$ in the standard form problem. $\square$

We now present a theorem that underlies a common method for finding solutions to linear programming problems.

9.4.2 The Fundamental Theorem of Linear Programming

THEOREM 9.2

An optimum solution to the linear programming problem (9.18) occurs at a point for which at most m_1 variables are positive and the remaining $n - m_1$ variables are zero. (The usual case is that exactly m_1 variables are nonzero.)

We give the proof of this theorem later, after we describe the simplex method for approximately solving (9.18).

DEFINITION 9.3 *A set of m_1 variables is called a basis. Points $x \in \mathbb{R}^n$ with $n - m_1$ components of x equal to zero and which satisfy the constraints $Ax = b$ of (9.19) are called basic feasible points or basic feasible solutions. Thus, a basic feasible solution consists of a nonnegative solution of m_1 variables to the m_1 linear equalities, with the remaining $n - m_1$ variables set to*

zero. (In general, a point satisfying all of the constraints of an optimization problem is called a feasible point, or feasible solution.⁶)

Theorem 9.2 suggests that one way of finding the optimum solution is by using a trial-and-error approach by selecting m_1 variables, setting the other variables to zero, and solving the resulting m_1 equations in m_1 variables. However, there are $n!/((n - m_1)!m_1!)$ ways of selecting m_1 variables from n variables; that is, there are $n!/((n - m_1)!m_1!)$ basic feasible points. (For example, if $m_1 = 6$ and $n = 15$, relatively small values, then there are $15!/(6!9!) = 5005$ possibilities. However, linear programming problems are solved today with m_1 and n in the hundreds of thousands, or more; a search of all possibilities would require an astronomical amount of time, even with today's most advanced computers.)

We now consider an efficient method of searching the basic feasible points, the well-known simplex method for linear programming.

9.4.3 The Simplex Method

The simplex method for finding solutions of (9.18) was invented in 1947 by George Dantzig. In the early days of computing (through the 1960's) it has been said that over half the time used on computer processors was spent performing the simplex method. The simplex method and its variants (including variants for structured problems, sparse constraint matrices A , etc.) are still extremely important.

REMARK 9.14 In general, the simplex method does not execute in polynomial time. That is, there is a sequence of problems with increasing n and m_1 such that the amount of work for the simplex method to complete the problem increases at a rate greater than $\mathcal{O}(n^k + m_1^k)$, for any integer k . In contrast, interior point methods, originating from Karmarkar's algorithm (see [42]) can be shown to execute in polynomial time, and, when implemented well, are practical for extremely large problems. Although thorough coverage of interior point methods is outside the scope of this book, a good introduction and overview can be found in [102]. In any case, variants of the simplex method are still highly practical for many problems, some with quite large n and m_1 , and some state-of-the-art software systems continue to use the simplex method. $\square$

In the simplex method, basic feasible solutions are tested one-by-one while steadily improving the objective function value, until the optimal value is

⁶A *feasible solution* is not to be confused with a solution to the optimization problem: A feasible solution is simply a solution of the system of constraints, without reference to the objective function.

reached. In fact, the simplex method can be viewed as a kind of steepest ascent method, proceeding from one extreme point of the feasible region to another, choosing to follow that edge of the simplex in which the objective is increasing most rapidly. We now present the simplex method informally.

Step 1: Find an initial basic feasible solution. One sure way to find an initial basic feasible solution is to introduce m_1 additional variables $x_{n+1}, x_{n+2}, \dots, x_{n+m_1}$ such that the linear equalities have the form

$$\begin{cases} a_{11}x_1 + a_{22}x_2 + \cdots + a_{1n}x_n + x_{n+1} & = b_1, \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + x_{n+2} & = b_2, \\ & \vdots \\ a_{m_1 1}x_1 + a_{m_1 2}x_2 + \cdots + a_{m_1 n}x_n + x_{n+m_1} & = b_{m_1}. \end{cases}$$

Then, an initial basic solution is

$$x_{n+1} = b_1, \quad x_{n+2} = b_2, \quad \dots, \quad x_{n+m_1} = b_{m_1},$$

with the other variables equal to zero. (In the simplex method, $x_{n+1}, x_{n+2}, \dots, x_{n+m_1}$ will eventually be set equal to zero.)

Step 2: Suppose that $x_1, \dots, x_{m_1}$ is a basic feasible solution; using elementary row operations (see Definition 3.12 on page 88), the linear system and objective function are put into the form

$$\begin{cases} x_1 & + & a'_{1,m_1+1}x_{m_1+1} + \cdots + & a'_{1,n}x_n = b'_1 \\ & x_2 & + & a'_{2,m_1+1}x_{m_1+1} + \cdots + & a'_{2,n}x_n = b'_2 \\ & \vdots & & & \\ & & x_{m_1} + & a'_{m_1,m_1+1}x_{m_1+1} + \cdots + & a'_{m_1,n}x_n = b'_{m_1} \\ \text{and} & & c'_{m_1+1}x_{m_1+1} + \cdots + & c'_n x_n = z - c'_0, \end{cases}$$

and the basic feasible solution is $x_1 = b'_1, x_2 = b'_2, \dots, x_{m_1} = b'_{m_1}$, with the other variables equal to zero.

Step 3 If $c'_j \leq 0$ for $j = m_1 + 1, \dots, n$, then the basic feasible solution is optimal, since z is decreased if $x_{m_1+1}, \dots, x_n$ are positive. Thus, the procedure is finished.

Step 4: If c'_j are not all ≤ 0 , then choose

$$c'_s = \max_{m_1+1 \leq j \leq n} c'_j.$$

This choice increases z the most for a unit change in x_s .

Step 5: The variable x_s will replace one of the variables x_1 to x_{m_1} . (In effect, we set one of x_1 through x_{m_1} to zero, and make x_s nonzero.) To

decide which variable to remove, note that each equation has the form $x_i + a'_{i,s}x_s = b'_i$. Thus, x_s is limited in magnitude by

$$\min_{\substack{1 \leq i \leq n \\ a'_{i,s} > 0}} \frac{b'_i}{a'_{i,s}}.$$

Let

$$\frac{b'_r}{a'_{r,s}} = \min_i \frac{b'_i}{a'_{i,s}}$$

for $a'_{i,s} > 0$. Then x_r is the variable to be removed.

(If $a'_{i,s} < 0$ for every i , then the optimal solution is unbounded. We give an example of this below.)

Step 6 Replace the form in Step 2 with

$$\left\{ \begin{array}{l} x_1 + a'_{1,r}x_r + a'_{1,m_1+1}x_{m_1+1} + \cdots + 0 + \cdots + a'_{1,n}x_n = b'_1 \\ x_2 + a'_{2,r}x_r + a'_{2,m_1+1}x_{m_1+1} + \cdots + 0 + \cdots + a'_{2,n}x_n = b'_2 \\ \vdots \\ a'_{r,r}x_r + a'_{r,m_1+1}x_{m_1+1} + \cdots + x_s + \cdots + a'_{r,n}x_n = b'_r \\ \vdots \\ x_{m_1} + a'_{m_1,r}x_r + a'_{m_1,m_1+1}x_{m_1+1} + \cdots + 0 + \cdots + a'_{m_1,n}x_n = b'_{m_1} \\ c_r^*x_r + c_{m_1+1}^*x_{m_1+1} + \cdots + 0 + \cdots + c_n^*x_n = z - c_0^* \end{array} \right.$$

That is, using $a'_{r,s}$ as a pivot element, we subtract multiples of the r -th row of the system of equations in step 2 to eliminate $a'_{i,s}$ from the i -th row, $1 \leq i \leq m_1 + 1$, $i \neq r$. Notice that $c_0^* = c'_0 + b'_r c'_s / a'_{r,s}$, so z is larger.

Example 9.6

Maximize $2x_1 + x_2 = z$ subject to

$$\left\{ \begin{array}{l} x_1 + 2x_2 \leq 10, \\ x_1 + x_2 \leq 6, \\ x_1 - x_2 \leq 2, \\ x_1 - 2x_2 \leq 1, \end{array} \right.$$

where $x_1 \geq 0, x_2 \geq 0$. First we put this problem into the following standard form:

$$\left\{ \begin{array}{ll} x_1 + 2x_2 + x_3 & = 10 \\ x_1 + x_2 + x_4 & = 6 \\ x_1 - x_2 + x_5 & = 2 \\ x_1 - 2x_2 + x_6 & = 1 \\ 2x_1 + x_2 & = z \end{array} \right.$$

1. An initial basic feasible set is $x_3 = 10$, $x_4 = 6$, $x_5 = 2$, $x_6 = 1$, $x_1 = 0$, $x_2 = 0$, and $z = 0$.
2. Choose $x_s = x_1$ and $x_r = x_6$ (add x_1 and remove x_6). The new system has the form:

$$\left\{ \begin{array}{rcl} 4x_2 + x_3 & - & x_6 = 9 \\ 3x_2 & + x_4 & - x_6 = 5 \\ x_2 & + x_5 & - x_6 = 1 \\ x_1 - 2x_2 & + & x_6 = 1 \\ 5x_2 & - & 2x_6 = z - 2 \end{array} \right.$$

The new basic feasible set is $x_1 = 1$, $x_3 = 9$, $x_4 = 5$, $x_5 = 1$, $x_2 = 0$, $x_6 = 0$, and $z = 2$.

3. Choose $x_s = x_2$ and $x_r = x_5$. The system then has the form:

$$\left\{ \begin{array}{rcl} x_3 & - & 4x_5 + 3x_6 = 5 \\ x_4 & - & 3x_5 + 2x_6 = 2 \\ x_2 & + & x_5 - x_6 = 1 \\ x_1 & - & 2x_5 - x_6 = 3 \\ & - & 5x_5 + 3x_6 = z - 7 \end{array} \right.$$

The new feasible set is $x_1 = 3$, $x_2 = 1$, $x_3 = 5$, $x_4 = 2$, $x_5 = 0$, $x_6 = 0$, and $z = 7$.

4. Choose $x_s = x_6$ and $x_r = x_4$. The resulting system then has the form:

$$\left\{ \begin{array}{rcl} x_3 - \frac{3}{2}x_4 + \frac{1}{2}x_5 & = & 2 \\ \frac{1}{2}x_4 - \frac{3}{2}x_5 + x_6 & = & 1 \\ x_2 + \frac{1}{2}x_4 - \frac{1}{2}x_5 & = & 2 \\ x_1 + \frac{1}{2}x_4 + \frac{1}{2}x_5 & = & 4 \\ -\frac{3}{2}x_4 - \frac{1}{2}x_5 & = & z - 10 \end{array} \right.$$

The new basic feasible set is $x_1 = 4$, $x_2 = 2$, $x_3 = 2$, $x_6 = 1$, and $z = 10$. This is the optimum value.

□

Notice that by brute force (trial-and-error), $\frac{6!}{3!3!} = 20$ linear systems would require testing. The simplex method only required 4.

9.4.4 Proof of the Fundamental Theorem of Linear Programming

PROOF The equality constraints in (9.18) can be written

$$v_1x_1 + v_2x_2 + \cdots + v_nx_n = b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{m_1} \end{pmatrix},$$

where $x_i \geq 0$ for $i = 1, \dots, n$, $\sum_{i=1}^n c_i x_i = z - c_0$, and $v_i = A_{:,i}$, $i = 1, \dots, n$ are m_1 -dimensional column vectors.

Assume that we have an optimal solution in which r variables, say the first r , are positive and $n - r$ variables are zero. If $r \leq m_1$, then the theorem is true, so let us assume $r > m_1$. Let $(x_1, x_2, \dots, x_r, x_{r+1}, \dots, x_n) = (\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_r, 0, 0, \dots, 0)$ be this solution. This means that

$$v_1\tilde{x}_1 + v_2\tilde{x}_2 + \cdots + v_r\tilde{x}_r = b, \quad \tilde{x}_1 > 0, \tilde{x}_2 > 0, \dots, \tilde{x}_r > 0,$$

and

$$c_1\tilde{x}_1 + c_2\tilde{x}_2 + \cdots + c_r\tilde{x}_r = \tilde{z}.$$

Since $r > m_1$, the vectors $v_1, v_2, \dots, v_r$ must be linearly dependent. It follows that there is a set of numbers, not all zero, such that $v_1y_1 + v_2y_2 + \cdots + v_ry_r = 0$, and we may assume that at least one $y_j > 0$ (otherwise we can multiply by (-1)).

Now, set $t = \max(y_j/\tilde{x}_j)$, a positive number. Then, we can show that

$$v_1\left(\tilde{x}_1 - \frac{y_1}{t}\right) + v_2\left(\tilde{x}_2 - \frac{y_2}{t}\right) + \cdots + v_r\left(\tilde{x}_r - \frac{y_r}{t}\right) = b.$$

That is, $\tilde{x}_j - y_j/t$ is also a solution. In addition, $\tilde{x}_j \geq y_j/t$ implies $\tilde{x}_j - y_j/t \geq 0$. Moreover, by the definition of t , at least one of these is zero. Thus, we have a feasible solution in which fewer than r are positive.

We now need to show that the solution is also optimal, i.e., that

$$c_1\left(\tilde{x}_1 - \frac{y_1}{t}\right) + c_2\left(\tilde{x}_2 - \frac{y_2}{t}\right) + \cdots + c_r\left(\tilde{x}_r - \frac{y_r}{t}\right) = c_1\tilde{x}_1 + c_2\tilde{x}_2 + \cdots + c_r\tilde{x}_r = \tilde{z},$$

which is true if $c_1y_1 + \cdots + c_ry_r = 0$. If this were not true, we could find u such that

$$u(c_1y_1 + \cdots + c_ry_r) = c_1(uy_1) + c_2(uy_2) + \cdots + c_r(uy_r) > 0.$$

Adding $\sum_{j=1}^r c_j\tilde{x}_j$ to both sides, we obtain

$$c_1(\tilde{x}_1 + uy_1) + c_2(\tilde{x}_2 + uy_2) + \cdots + c_r(\tilde{x}_r + uy_r) > c_1\tilde{x}_1 + c_2\tilde{x}_2 + \cdots + c_r\tilde{x}_r = \tilde{z}.$$

But $\tilde{x}_j + uy_j$, $1 \leq j \leq r$, is easily shown to be a solution for any u . By making u sufficiently small, it would be a nonnegative solution. But this would mean that $\tilde{x}_j + uy_j$ would give a larger value of z than the $\tilde{x}_j$, which is a contradiction.

We have thus proved that the number of positive variables in an optimal solution can be reduced if $r > m_1$. Thus, we are led to a solution in which at most m_1 variables are positive. $\square$

9.4.5 Degenerate Cases

Although the Fundamental Theorem of Linear Programming states that a maximum value of the objective function $-\varphi(x) = c_0 + \sum_{j=1}^n c_j x_j$ occurs at a point for which at least m_1 of the x_i are nonzero, it does not rule out other points at which $-\varphi(x)$ is maximum. That is, there may be more than one point at which $-\varphi$ obtains its (unique) maximum value.

Example 9.7

Minimize	$10A + 3.5B + 4C + 3.2D$
Subject to:	
	$100A + 50B + 80C + 40D \leq 200,000,$
	$12A + 4B + 4.8C + 4D \geq 18,000,$
	$0 \leq 100A \leq 100000,$
	$0 \leq 50B \leq 100000,$
	$0 \leq 80C \leq 100000,$
	$0 \leq 40D \leq 100000,$

Any point along the portion of the line $B = 0$, $D = 2500$, A and C given parametrically by

$$\begin{pmatrix} A \\ C \end{pmatrix} \approx \begin{pmatrix} 666.\bar{6} \\ 0 \end{pmatrix} + t \begin{pmatrix} -0.3714 \\ 0.9285 \end{pmatrix},$$

with $0 \leq C \leq 1250$, is a solution to this problem. $\square$

Such degenerate problems are relatively common in practice, and occur when the vectors formed from the coefficients of the objective function and the vectors formed from the coefficients of the constraints are linearly dependent. Some software for solving linear programs removes some linear dependence by preprocessing, prior to applying the simplex method or interior point method, but linear dependence such as in Example 9.7 is intrinsic to the problem. Many software systems will return the optimum value for $-\varphi$ and an optimizing point that happens to be at a vertex, without indicating that this optimizing point is not unique.

9.4.6 Failure of Linear Programming Solvers

Linear programming solvers can fail, due both to intrinsic properties of the problem posed to them and also due to numerical issues, such as roundoff error or efficiency considerations. There are two types of problems whose formulations do not admit solutions.

- *unbounded problems*, and
- *infeasible problems*.

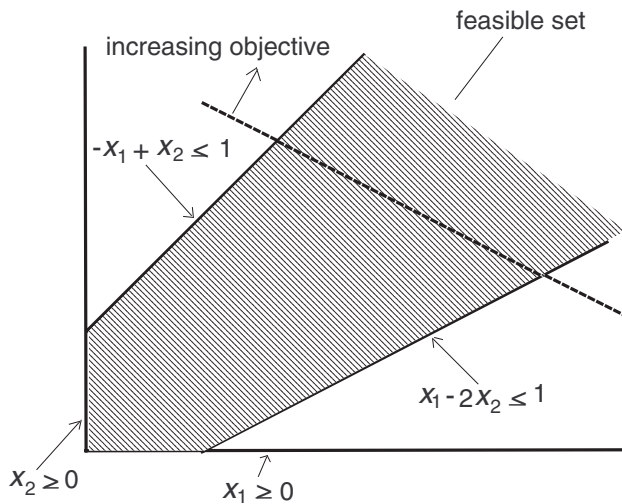


FIGURE 9.4: Graph of the constraint set and the objective function z for Example 9.8, an unbounded LP.

Example 9.8

(Illustration of an unbounded linear program) Consider maximizing $x_1 + 2x_2$ subject to

$$-x_1 + x_2 \leq 1, \quad x_1 - 2x_2 \leq 1, \quad x_1 \geq 0, \quad x_2 \geq 0.$$

The feasible set is the shaded region in Figure 9.4, while a level curve of the objective is given by the dashed line. Observe that one can increase the objective without bound by choosing only points within the feasible set. $\square$

REMARK 9.15 The feasible set must be unbounded for a linear program to be unbounded,⁷ but the converse is not true. In particular if, in Example 9.8, the objective were to minimize $x_1 + 2x_2$ rather than maximize $x_1 + 2x_2$, then the problem would have a solution at $x_1 = 0$, $x_2 = 0$, and the simplex method would find it. $\square$

Example 9.9

(Example of an infeasible linear program) Consider maximizing $x_1 + x_2$ subject to

$$x_1 + x_2 \leq 1, \quad 2x_1 - x_2 \leq -4, \quad x_1 \geq 0, \quad x_2 \geq 0.$$

Then there are no points that satisfy all of the constraints, so the feasible set is empty. $\square$

REMARK 9.16 The notions of unbounded problems and infeasible problems carry over to the general nonlinear optimization problem (9.1), but the ways that such problems can occur are more complicated in the general case. $\square$

One reference with examples and references to problems that can occur due to roundoff error in linear programming solvers is [63].

9.4.7 Software for Linear Programming

Due to the commercial value of linear programming software, many of the most efficient and reliable linear programming packages are at present proprietary, available for licensing fees. Some of these are in add-on packages to interactive systems, such as `linprog` in MATLAB's optimization toolbox. However, there exist various reasonably competitive free packages, such as CLP from the Computational Infrastructure for Operations Research (COIN-OR) project (see <http://www.coin-or.org/>).

9.4.8 Quadratic Programming and Convex Programming

Quadratic programs are instances of the general optimization problem (9.1) in which the objective φ is quadratic and the constraints c and g are linear. That is, φ is of the form

$$\varphi(x) = \frac{1}{2}x^T Hx + h^T x + d, \tag{9.20}$$

⁷One way of seeing this is to remember the general theorem that says that a continuous function over a compact set must attain its maximum and minimum on that set.

for some symmetric matrix H . If the Hessian matrix H of φ is positive definite, then the quadratic programming problem is tractable, the problem can be posed in terms of a linear program.

There are special algorithms for quadratic programming, such as **quadprog** from the MATLAB optimization toolbox.

Increasingly, software is also becoming available for *convex programs*. These are instances of (9.1) in which the function φ and the functions g_i are convex, and there are either no equality constraints, or else the equality c_i constraints are linear.

9.5 Dynamic Programming

Dynamic programming is a technique for solving a special class of optimization problems called *multi-stage decision processes* [54, 65]. It is difficult to present a specific mathematical form for the class of optimization problems which dynamic programming can solve. Dynamic programming is a computational technique that generally is used to reduce a difficult problem in n variables into a series of optimization problems in one variable through application of recurrence relations. (The whole method might well have been named recurrence optimization.) The possibility of applying dynamic programming depends on a successful formulation of the problem in terms of a multi-stage decision process. Two examples are presented here that illustrate the technique.

Example 9.10

Suppose that you own a lake and each year you either fish and remove 70% of the fish, or you do not fish. If you fish and remove 70% of the fish, the fish that were not caught reproduce, replenishing the original population. If you don't fish, the fish population doubles in size the next year. The initial fish population is 10 tons. Your profit is \$1000/ton and the interest rate is constant at 25%. You have 3 seasons to fish. What procedure will optimize your profit?

A schematic diagram of the fish population is given in Figure 9.5. A schematic diagram of present values of profit in \$1000's is given in Figure 9.6. (Recall that the present value at time 0 is equal to the amount of money at the n -th year divided by $(1 + i)^n$.) The optimum profit at each node is obtained by working backward. The optimum strategy is indicated by the darkened path. $\square$

Dynamic programming solves the problem step-by-step, starting at the termination time and working back to the beginning. For this reason, dynamic

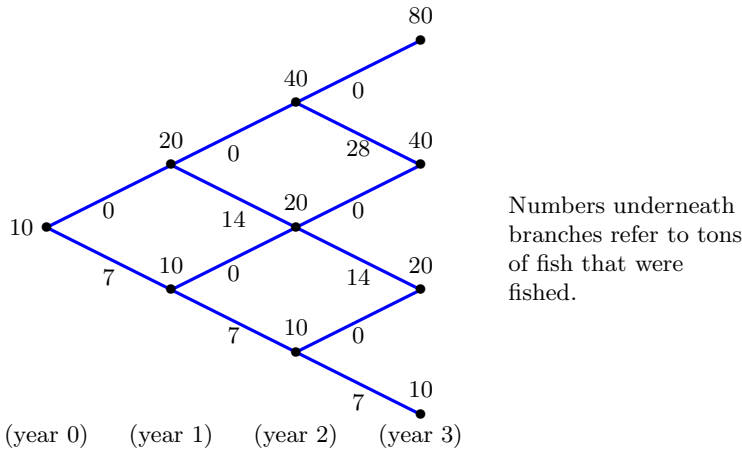


FIGURE 9.5: Diagram of the fish population for Example 9.10 of a multi-stage decision process.

programming is sometimes characterized by the phrase “it solves the problem backward.” The next example will make this clearer.

Example 9.11

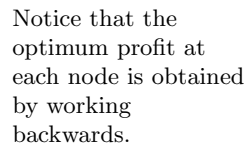
Suppose that you own a gold mine. You wish to operate the mine for ten years. The cost each year to extract z ounces of gold is $\$500z^2/x$ where x is the number of ounces remaining at the beginning of the year. Assume that the price of gold is $\$400/\text{oz} = g$, the interest rate is 10%, and $x_0 = 50,000$ ounces. What is the maximum amount of profit?

Let

- $V_j = \left\{ \begin{array}{l} \text{the value of the mine from the } j\text{-th year to the last year, adjusted} \\ \text{according to the 10\% interest rate to be in dollars at the beginning} \\ \text{of the 10 years.} \end{array} \right\}$
- $x_j = \{\text{number of ounces of gold remaining at the beginning of the } j\text{-th year},\}$
- $g = \{\text{the price of gold (assumed to be constant)},\}$
- $z_j = \{\text{number of ounces of gold extracted in } j\text{-th year}\}, \text{ and}$
- $d = \{\text{the discount factor}\} = 1/1.1.$

We consider first the last year. Assuming no gold is left at the end of the 10-th year, we have

$$V_9 = \max_{0 \leq z_9 \leq x_9} \{gz_9 - 500z_9^2/x_9\} = \frac{g^2x_9}{2000} = k_9x_9,$$



where $k_9 = g^2/2000$. Now, considering the previous year, we have $x_9 = x_8 - z_8$, and

In general, for the j -th year, we have $x_{j+1} = x_j - z_j$, and

that is,

Thus, $V_0 = k_0 x_0$ is the optimal value of the mine, and k_0 can be determined by working backwards. Indeed, we have the following table.

i	k_i
9	80.00
8	126.28
7	155.47
6	174.79
5	187.96
4	197.13
3	203.58
2	208.17
1	211.45
0	213.81

Hence, $V_0 \approx \$213.81 \times 50,000 = \$10,691,000$. Also, we may find the optimum mining strategy. In particular, z_j is the maximizer in the j -th subproblem, namely,

$$z_j = \frac{(g - dk_{j+1})x_j}{1000}, \quad 1 \leq j \leq 8, \quad \text{with} \quad z_9 = \frac{gx_9}{1000},$$

so

$$x_{j+1} = x_j - z_j = \left[1 - \frac{g - dk_{j+1}}{1000} \right] x_j,$$

with $x_0 = 50000$.

□

9.6 Global (Nonconvex) Optimization

Global optimization, in contrast to local optimization, is the process of finding the minimum of φ in problem (9.1) over the entire feasible set, if such a minimum exists. As discussed in the introduction to this chapter, there are no known algorithms that are guaranteed to find approximate solutions to all global optimization problems with n variables in $\mathcal{O}(n^k)$ time, for any integer k . In fact, present algorithms that are guaranteed to find a global optimum for the general case⁸ of problem (9.1) subdivide the region in a technique akin to adaptive quadrature.

Nonetheless, global optimization problems are important in applications, and research on efficient algorithms has exploded during the past several decades. Such algorithms can be classified in various ways. One classification is as follows.

⁸that is, without further assumptions on φ , c , and g , other than perhaps differentiability

deterministic algorithms: These are algorithms that are certain to find the global optimum and global optimizers to within the specified accuracy provided that no roundoff errors are made and the algorithms continue to completion.

heuristic algorithms: These algorithms, which can include statistical techniques, may work well for many problems, but there is no guarantee that the answers they provide are close to an actual global minimum.

Among deterministic algorithms, the algorithms can

- attempt to find merely an approximation to the global optimum and a single set of parameters x corresponding to the global optimum;
- attempt to find not only an approximation to the global optimum, but also approximations to all optimizing points⁹ x . Such algorithms are often called *complete search algorithms*.

Deterministic algorithms may also be *automatically verified*, in which, if the algorithm finishes, it is guaranteed¹⁰ to return mathematically rigorous bounds (given by machine-representable numbers) on the global optimum and the global optimizing points; in automatically verified complete search algorithms, the algorithm (if it completes) is guaranteed to supply mathematically rigorous bounds on all of the coordinates of all points $x \in \mathbb{R}^n$ with $\varphi(x) = \varphi^*$.

As mentioned earlier in this chapter, there are classes of problems with relatively fast deterministic algorithms. These include

- linear programs;
- quadratic programs;
- convex optimization problems;¹¹
- problems with special structure from various important application areas.

There are also important problem classes with special structure, but for which fast deterministic algorithms are not known in general. An example of this is

integer programming problems: In these problems, some or all of the variables x_i are constrained to be integers. (In principle, such constraints can be made to fit into the general form (9.1) (on page 487) by appending constraints of the form $c_i(x) = \sin(\pi x_i) = 0$. However, special techniques are often employed for integer programming problems.)

⁹Even though the global optimum φ^* must be unique, it may be that $\varphi(x) = \varphi^*$ at many points x .

¹⁰unless there is a programming error or a computer hardware malfunction

¹¹in which the objective and all of the constraint functions are convex

Heuristic algorithms take various forms. Several important classes of heuristic algorithms include a statistical component.

In the remainder of this section, we briefly introduce several of the more prominent current general techniques, both heuristic and deterministic, not verified and automatically verified, for global optimization.

9.6.1 Genetic Algorithms

Genetic algorithms are a heuristic global optimization technique with a random component [11, 38]. Genetic algorithms (which we abbreviate “GA’s”) are modeled after genetics and evolution. GA’s are designed to find low objective values for large, complex problems. The search proceeds in a survival-of-the-fittest manner. The evolution of a population of potential solutions is monitored until the superior parameter values dominate the population.

The starting point in using GA’s is by representing the problem in a “biological” manner. This often leads to a binary representation, that is, representing the parameters of the problem using a string of binary digits. Perhaps the best way to understand GA’s is by considering a simple problem.

Example 9.12

Consider the problem of finding $0 \leq x \leq 63$ that minimizes

$$\varphi(x) = x^2 - 60x - 100. \quad (9.21)$$

The solution to this problem is $x = 30$. We will represent this problem in binary form as finding $000000 \leq x \leq 111111$ so that $\varphi(x)$ is minimum.

The solution is $x = 011110$. This problem has a biological parallel. The bit string can be viewed as a chromosome-type structure. The 0’s and 1’s correspond to genes in the chromosome. Just as a chromosome can be decoded to reveal characteristics of an individual organism, a bit string can be decoded to reveal a potential solution. $\square$

A genetic algorithm procedure consists of the following steps. Each step will be illustrated with Example 9.12.

Step 1: An initial population of potential solutions is randomly created using a random number generator. Suppose that our initial randomly generated population consists of the four strings:

000001, 110100, 111000, 010100 in decimal: 1, 52, 56, 20.

Step 2: Calculate the performance or fitness of each individual in the population. GA’s are typically implemented in such a way that performance, or fitness, is the value to be minimized.

For our problem, a good measure of fitness is $\varphi(x)$. The lower $\varphi(x)$ is, the more fit string x is to survive and to breed. For the initial population, we have the table:

String x	Decimal equivalent	Fitness $\varphi(x)$
000001	1	-159 (least fit)
110100	52	-516
111000	56	-324
010100	20	-900 (most fit)

Step 3: Select individuals to be parents for the next generation. Basically, better performing individuals are chosen as parents. Poorly performing individuals do not produce offspring. One possible scheme for selecting parents is by throwing away the worst-performing string and replacing it with the best-performing string. Applying this procedure to our example yields the strings:

010100, 110100, 111000, 010100.

Step 4: Create a second generation of children from the parents. There are a variety of genetic operations that can be performed at this stage. One of the most powerful methods is crossover, and is motivated by an analogous biological process. In crossover, two strings are randomly chosen. A point is randomly chosen at which the two strings are to be cut. Another random number is then chosen to indicate whether or not crossover is to be performed. If crossover is to be performed (perhaps set for 60% of the time), then the tails and heads of the two strings are exchanged. Notice in the crossover process, several random selections were performed. A pair was randomly chosen, where to cut the strings was randomly decided, and whether or not to perform the crossover was randomly decided. Crossover is a powerful process that extends the search in many directions.

Consider our example. Suppose crossover is decided for 010100 and 111000. They are cut at 0 – 10100 and 1 – 11000. Crossover is performed to yield the new population: 011000, 110100, 110100, 010100.

Mutation is a second possible genetic operation that can be performed. (This is also inspired by a biological operation.) In mutation, a particular bit in a particular string is randomly selected and changed from 0 to 1 or from 1 to 0. Generally, mutation is performed infrequently.

Crossover and mutation are common genetic operations used in genetic algorithms. Crossover tends to pass on attractive patterns from one generation to the next. Mutation gently nudges the search in slightly different directions.

Step 5: Steps 2 through 4 are repeated until the population converges. The parent selection scheme ensures that good strings drive out bad strings. Crossover spreads pieces of well-performing strings to poor-performing strings. As these steps are repeated over and over, the population gradually becomes more homogeneous. When all strings are identical, the population is said to be fully converged.

Consider now two more generations of our example.

Step 2: (Second generation)

String x	Decimal equivalent	Fitness $\varphi(x)$
011000	24	-964
110100	52	-516
110100	52	-516
010100	20	-900

Step 3: 110100 is replaced by 011000 to yield the population:

011000, 011000, 110100, 010100.

Step 4: Crossover is performed with strings 011000 and 010100. The strings are cut at 011 – 000 and 010 – 100. The new population is 011000, 010000, 110100, 011100. A mutation is made to the first string in the fifth position. The new population is 011010, 010000, 110100, 011100.

Step 2: (Third generation)

String x	Decimal equivalent	Fitness $\varphi(x)$
011010	26	-984
010000	16	-804
110100	52	-516
011100	28	-996

Step 3: 110100 is replaced by 011100 to yield the population:

011010, 010000, 011100, 011100.

Step 4: Crossover is performed with strings 011010 and 011100. The strings are cut at 0110 – 10 and 0111 – 00. The new population is

011110, 010000, 011100, 011000.

Step 5: (Fourth generation)

String x	Decimal equivalent	Fitness $\varphi(x)$
011110	30	-1000
010000	16	-804
011100	28	-996
011000	24	-964

Notice that the fourth generation is much more fit than the first generation. Much more complicated problems can be approximately solved using a genetic algorithm procedure.

9.6.2 Simulated Annealing

Simulated annealing is another heuristic technique with a random component. However, as the name implies, simulated annealing is modeled after the annealing process in metallurgy. In addition to web resources for simulated annealing, the monograph [95] provides an introduction.

9.6.3 Branch and Bound Algorithms

In branch and bound algorithms, the entire domain is systematically searched, eliminating portions of the domain that cannot contain optimizers.

9.6.3.1 Branch and Bound Procedures in General

Branch and bound processes are a general class of deterministic algorithms that proceed by the following steps.

1. An upper bound $\overline{\varphi}$ on the global optimum over the feasible set (defined by the constraints) is established.
2. The initial region D is subdivided into two or more subregions $\tilde{D}$. (This is the *branching* step, where we branch into subregions.)
3. The range of the objective function is bounded below over each subregion $\tilde{D}$, to obtain

$$\underline{\varphi}(\tilde{D}) \leq \{\varphi(x) \mid x \in D, c(x) = 0, g(x) \leq 0\}.$$

(This is the *bounding* step, where we bound the range of φ .)

4. IF $\underline{\varphi} > \overline{\varphi}$ THEN

$\tilde{D}$ is discarded,

ELSE IF the diameter of $\tilde{D}$ is smaller than a specified tolerance THEN

$\tilde{D}$ is put onto a list of boxes containing possible global optimizers.

ELSE

$\tilde{D}$ is put onto a list for further branching and bounding through steps 2 and 3.

END IF

The upper bound $\bar{\varphi}$ on the global optimum is sharpened, often with consideration of each new region $\tilde{D}$, for example, by evaluating φ at a point $x \in \tilde{D}$ that is known to be feasible.

Some of the ways the bounding process can be done are

- by using Lipschitz constants to bound φ , c , and g ,
- by interval arithmetic, or
- more generally, by *relaxations*.

DEFINITION 9.4 A relaxation of the global optimization problem (9.1) is a related problem, where φ is replaced by some related function $\check{\varphi}$, each set of constraints is replaced by a related set of constraints, such that, if φ^* is the global optimum to the original problem (9.1) and $\varphi^\dagger$ is the global optimum to the relaxed problem, then $\varphi^\dagger \leq \varphi^*$.

REMARK 9.17 In particular:

- if one deletes one or more constraints, one obtains a relaxation.
- If one:
 1. replaces φ over $\tilde{D}$ by $\check{\varphi}$ such that $\check{\varphi}(x) \leq \varphi(x) \forall x \in \tilde{D}$, and/or
 2. replaces one or more $c_i(x) = 0$ by a pair $\{c_i(x) \leq 0, -c_i(x) \leq 0\}$ then replaces $c_i(x) \leq 0$ by $\check{c}_i(x) \leq 0$, where $\check{c}_i(x) \leq c_i(x)$, $\forall x \in \tilde{D}$, and/or replaces $-c_i(x) \leq 0$ by $-\check{c}_i(x) \leq 0$, where $-\check{c}_i(x) \leq -c_i(x)$, $\forall x \in \tilde{D}$, and/or
 3. replaces one or more $g_i(x) = 0$ by $\check{g}_i(x) \leq 0$, where $\check{g}_i(x) \leq g_i(x)$, $\forall x \in \tilde{D}$,

then the resulting problem is a relaxation to the original problem.

□

An important type of problem to which relaxations typically are applied is integer programming problems. A natural relaxation for such problems is to ignore the constraint that a variable x_i be integer, and treat x_i as a real variable.

Example 9.13

Let the global optimization problem be defined by

$$\begin{aligned} \varphi(x_1, x_2) &= 2x_1^2 - 2x_1x_2 + x_2^2 - 4x_1 + 4, \\ \text{subject to } &1 \leq x_1 \leq 11, \quad 1 \leq x_2 \leq 11, \quad x_1 \text{ and } x_2 \text{ integers.} \end{aligned}$$

This integer programming problem has global optimum $\varphi^* = 0$ at the unique global optimizer $x_1 = x_2 = 2$. $\square$

We may apply a relaxation to Example 9.13 by assuming the variables are real. To obtain an initial $\overline{\varphi}$, we use a local optimizer to obtain a point $\tilde{x}$, possibly adjusting $\tilde{x}$ so that it has integer coordinates, then taking $\overline{\varphi} = \varphi(\tilde{x})$. For illustration purposes, suppose we have obtained $\tilde{x} = (1, 1)$, so $\overline{\varphi} = \varphi(1, 1) = 1$. Suppose also that we start with the initial region

$$D = \mathbf{x} = ([1, 11], [1, 11])^T,$$

and we subdivide (“branch”) on $\mathbf{x}$ by bisecting the widest coordinate of $\mathbf{x}$. The branch and bound algorithm would then proceed as follows.

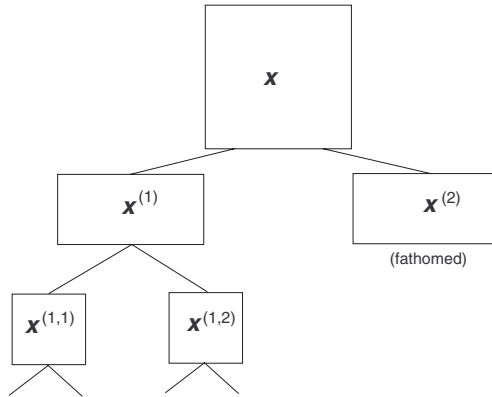


FIGURE 9.7: The search tree for Example 9.13

Step 2: Cut $\mathbf{x}$ into

$$\mathbf{x}^{(1)} = ([1, 11], [1, 6])^T \quad \text{and} \quad \mathbf{x}^{(2)} = ([1, 11], [6, 11])^T.$$

Step 3: Here, for simplicity, we will use interval arithmetic to bound φ , although there are many alternative techniques, such as the ones we explain in Section 9.6.3.2 below. For illustration purposes,¹² we rewrite φ

¹²to avoid obscuring the process by having to do too many steps

here as

$$\varphi(x) = (x_1 - x_2)^2 + (x_2 - 2)^2.$$

(In this form, there is less overestimation of the range in the interval evaluations.) Using interval arithmetic on $\mathbf{x}^{(1)}$, we obtain $\varphi(\mathbf{x}^{(1)}) \subseteq [0, 116]$.

Step 4: Since $\overline{\varphi} \in [0, 116]$, we cannot delete $\mathbf{x}^{(1)}$; therefore, we place $\mathbf{x}^{(1)}$ on a list $\mathcal{L}$ to be processed further in Step 2.

Step 3: Now working on $\mathbf{x}^{(2)}$, we obtain $\varphi(\mathbf{x}^{(2)}) \subseteq [16, 181]$.

Step 4: Since $\overline{\varphi} \notin [16, 181]$, we may remove $\mathbf{x}^{(2)}$ from further processing. We say that $\mathbf{x}^{(2)}$ has been *fathomed*.

Step 2: Now, $\mathbf{x}^{(1)}$ is at the top of the list for further processing.¹³ In that case, we form

$$\mathbf{x}^{(1,1)} = ([1, 6], [1, 6])^T \quad \text{and} \quad \mathbf{x}^{(1,2)} = ([6, 11], [1, 6])^T.$$

Step 3: We have $\varphi(\mathbf{x}^{(1,1)}) \subseteq [0, 41]$.

Step 4: Since $\overline{\varphi} \in [0, 41]$, we must put $\mathbf{x}^{(1,1)}$ onto $\mathcal{L}$ for further processing.

Step 3: We have $\varphi(\mathbf{x}^{(1,2)}) \subseteq [0, 116]$. Since $\overline{\varphi} \in [0, 116]$, we must also store $\mathbf{x}^{(1,2)}$ in the list.

This process can be depicted by a *search tree*, as is illustrated in Figure 9.7.

Branch and bound processes differ widely in their ability to solve problems efficiently. Items affecting efficiency include

- the techniques used to get upper bounds $\overline{\varphi}$ on the global optimum;
- the techniques used to obtain lower bounds on the objective over subregions of the domain D ;
- the way that the list $\mathcal{L}$ of boxes that haven't yet been fathomed is ordered;
- acceleration procedures, used to eliminate a subregion or reduce its size prior to subdivision;
- the way that a region is subdivided (into how many subregions, where the cuts are made, etc.).

¹³There are various methods for ordering the list $\mathcal{L}$.

9.6.3.2 Some Special Relaxations

A common type of relaxation is a *linear relaxation*, in which the objective and constraints are replaced by linear functions. A big advantage of this is that well-developed linear programming technology, more tractable and better understood than general nonlinear optimization, can be used to find solutions to the relaxations. Thus, this kind of relaxation is commonly used in software that does not employ interval evaluations. Various techniques can be used to obtain such relaxations; some of these are general and can be done automatically by the machine. An example of such a technique appears in [47]. Such automatic creation of relaxations can proceed by decomposition into elementary operations, as in automatic differentiation (see page 329), or by more sophisticated techniques. Such a decomposition technique is illustrated in [47].

Convex relations are also sometimes used.

9.6.3.3 Acceleration Procedures

The basic process in branch and bound algorithms is to bound the optimum above, bound the ranges of the objective and constraints over subregions, then reject those subregions over which the range bounds show that the problem either must be infeasible or the objective function must be greater than the upper bound on the optimum. Although practical for some problems, this simple basic technique requires much computation, especially in higher dimensions, where, with n variables, bisecting uniformly so the resulting box has half the diameter of the original box results in 2^n sub-boxes. For this reason, various other techniques are used in practical software systems to reduce the volume of region to be searched. We now mention a few of these.

Use of Local Optimization Software. Local optimization software generally completes its computations much more rapidly than a branch and bound procedure. If such software provides a point $\hat{x}$ such that $\hat{x}$ is feasible, then $\overline{\varphi}$ may be replaced by the minimum of its previous value and $\varphi(\hat{x})$. It is likely that such $\hat{x}$ will have lower values of φ than $\varphi(x)$, where x is randomly chosen. Furthermore, for constrained problems, it is only valid to use values $\varphi(x)$ if x is feasible, and local optimization software will usually converge to such feasible points, and often converges to a point near a global optimizing point. Lower values for $\overline{\varphi}$, especially when found early in the branch and bound process, lead to earlier rejection of larger sub-regions, reducing the need for further branching.

Constraint Propagation. A simplistic view of constraint propagation is in terms of solving the nonlinear relation representing $c_i = 0$ or $g_i \leq 0$, or a component of the Kuhn–Tucker conditions for one variable, then comput-

ing new bounds on that variable based on the present bounds on the other variables, similar to the nonlinear Jacobi method we explained on page 461.

Example 9.14

Consider

$$\begin{aligned} &\text{minimize } \varphi(x) = x_1^2 - x_2^2 \\ &\text{subject to } x_1^2 + x_2^2 = 1, \\ &\quad x_1 + x_2 \leq 0. \end{aligned}$$

Suppose the goal is to find *all* optimizing points, and suppose we have used a local optimizer to find the feasible point $\hat{x} = (x_1, x_2) = (0, -1)$, with $\varphi(\hat{x}) = -1 = \bar{\varphi}$, and suppose we are searching in the initial box $(x_1, x_2) \in ([-1, 1], [-1, 1])$. Then, in addition to the constraints, we have the condition

$$\begin{aligned} x_1^2 - x_2^2 \leq -1 &\Leftrightarrow \left\{ x_1 \leq \sqrt{x_2^2 - 1} \text{ and } x_1 \geq -\sqrt{x_2^2 - 1} \right\} \\ &\Leftrightarrow \left\{ x_2 \geq \sqrt{x_1^2 + 1} \text{ or } x_2 \leq -\sqrt{x_1^2 + 1} \right\}. \end{aligned}$$

That is, we can use the upper bound $\bar{\varphi}$ on the global optimum and to either reduce the range of x_1 , given a range on x_2 , or to reduce the range of x_2 , given a range on x_1 . Taking $x_2 \in [-1, 1]$ and substituting into the inequalities for x_1 , we obtain

$$x_1 \leq \sqrt{[-1, 1]^2 - 1} = \sqrt{[0, 1] - 1} = \sqrt{[-1, 0]}.$$

Although there are values $x \in [-1, 0]$ for which $\sqrt{x}$ is not defined as a real number, in this context, it is appropriate to interpret $\sqrt{[-1, 0]}$ to mean a bound on the range of $\sqrt{\cdot}$ over those values of $x \in [-1, 1]$ at which $\sqrt{x}$ is defined.¹⁴ Interpreted this way, $\sqrt{[-1, 0]} = [0, 0]$, so

$$x_1 \leq 0.$$

Taking the other condition $x_1 \geq -\sqrt{x_2^2 - 1}$ similarly gives $x_1 \geq 0$, and combining gives $x_1 = 0$. Now, solving for x_2 in the equality constraint $x_1^2 + x_2^2 = 1$ gives

$$x_2 = \sqrt{1 - x_1^2} \quad \text{or} \quad x_2 = -\sqrt{1 - x_1^2}.$$

Since the previous computation established that $x_1 \in [0, 0] = 0$, we obtain $x_2 = 1$ or $x_2 = -1$, giving only two points $(0, -1)$ and $(0, 1)$. Substituting the second point into the inequality constraint leads to a contradiction,

¹⁴Some systems for interval arithmetic do interpret interval values of $\sqrt{\cdot}$ in this way, while others do not. In the context of interval Newton methods, it is important that computation not continue when part of the argument is out of range. INTLAB interprets $\sqrt{[-1, 0]}$ as a set of numbers in the complex plane, which is appropriate in yet other contexts.

thus proving that $(x_1, x_2) = (0, -1)$ is the only possible optimizing point for the problem within the box $([-1, 1], [-1, 1])$, without any need for further branching in the branch and bound process. $\square$

Constraint propagation has numerous variants, depending on which relations are solved for which variables, and depending on whether or not the system of constraints is modified. Computer languages have been developed specifically for constraint propagation, also called *constraint programming*. One reference on the subject is [5].

Techniques for Sharper Lower Bounds on the Range. A commonly used technique is to replace the optimization problem (9.1) by a linear relaxation, resulting in a linear program. If the original optimization problem is convex, then the problem can be approximated arbitrarily well by a linear relaxation, and the lower bound for φ over a particular region can be computed sharply by solving a linear program.¹⁵ Although nonconvex nonlinear programs cannot be approximated arbitrarily closely by linear programs, relaxations can still be obtained, and these relaxations can give better lower bounds on the range of φ than, say, a naive evaluation using interval arithmetic. One way of viewing the reason for this is that by simply evaluating φ , we do not take into account that a portion of the range is over points that are infeasible, whereas the solution to a linear relaxation does include the constraints.

Linear relaxations can be used in various ways other than for determining lower bounds on the range of φ . Some of these ways are described in [93], although additional research has been done on the subject since then.

Interval Newton Methods. Proposition 8.2 (page 464) states that any solution of $F(x) = 0$ that are in $\mathbf{x}$ must also be in the image of $\mathbf{x}$ under an interval Newton method, while Theorem 8.9 (page 465) states that, if a Lipschitz matrix is used in the interval Newton method and the interval Newton method maps $\mathbf{x}$ into the interior of $\mathbf{x}$, then there is a unique solution to $F(x) = 0$ within $\mathbf{x}$. These facts can be used in a branch and bound algorithm: Instead of subdividing a box $\mathbf{x}$, an interval Newton method can reduce the volume of $\mathbf{x}$ through an iterative process, with a guarantee that no critical points of the optimization problem are lost.¹⁶ This is effective for some problems, but tends to be effective primarily when the Jacobian matrix for F is well-conditioned. In fact, however, the Jacobian matrix for the Kuhn–Tucker conditions (9.17) typically contains singular matrices unless

¹⁵which may be a large, sparse linear program

¹⁶Here, $F(x) = \nabla \varphi$ if the problem is unconstrained, and $F(x)$ is defined to be the Kuhn–Tucker function (9.17) or the Fritz John function for constrained problems.

fairly sharp bounds on the Lagrange multipliers u and v are known. Generally, interval Newton methods are most practical in finding sharp bounds in which it is proven that a solution exists, provided a good approximate solution is already known, while interval Newton methods are not as effective in narrowing wide bounds.

Explicit Constructions. A phenomenon called the *clustering effect* typically occurs in the basic branch and bound algorithm. Because $\overline{\varphi}$ can be higher (however slightly) than the actual global minimum, and because the lower bounds $\underline{\varphi}(\mathbf{x})$ on the range of φ over boxes $\mathbf{x}$ that are adjacent to boxes that contain the global minimum is less than the exact lower bound on the range, and since the exact lower bound on the range of φ is near the global optimum because of the continuity of φ and the fact that $\mathbf{x}$ is near an optimizing point, it can happen that $\underline{\varphi}(\mathbf{x}) < \overline{\varphi}$, even though $\mathbf{x}$ cannot contain any global optimizing points.¹⁷ The result is that the algorithm produces clusters of boxes around optimizing points, which under certain conditions, are even larger if the stopping tolerance (giving a box size for which the box is no longer subdivided) in the branch and bound algorithm is made smaller. This phenomenon was first analyzed mathematically in [26, 46], and later in [79].

The following procedure can ameliorate the clustering problem.

1. Assume that an approximate optimizing point $\tilde{x}$ has been found.
2. Construct a small box $\tilde{\mathbf{x}}$ centered about $\tilde{x}$, such that $\tilde{\mathbf{x}}$ has diameter an order of magnitude larger than the box diameter tolerance for the branch and bound algorithm.
3. Eliminate $\tilde{\mathbf{x}}$ from the search region.

A similar technique appeared in a general setting in [43], while an explicit method for eliminating $\tilde{\mathbf{x}}$ from the search region is explained in [44, §4.3.1].

Interval Newton methods can also help the clustering problem, when they are applicable.

9.6.3.4 Considerations for Automatically Verified Algorithms

Interval arithmetic is commonly used in commercial software¹⁸ employing branch and bound methods for global optimization, to economically compute bounds on ranges and in constraint propagation processes. However, such software usually does not claim to find mathematically rigorous bounds on all optimizing points, and special care must be taken in those cases.

¹⁷This is also true in constrained problems, where $\mathbf{x}$ can be near the feasible set, even though it does not contain any feasible points.

¹⁸such as BARON [93]

For example, values of $\bar{\varphi}$ can be obtained in constrained problems by evaluating φ at feasible points x . In algorithms that do not take account of roundoff error and other computational errors (such as terminating an iteration when $|x_{k+1} - x_k|$ is small but not equal to zero), it is permissible to set $\bar{\varphi}$ to $\hat{x}$, where $\hat{x}$ is approximately feasible. Such $\bar{\varphi}$ are usually close to an upper bound on the global optimum, but may be slightly less than the actual upper bound. In most cases, branch and bound algorithms using such approximate $\bar{\varphi}$ will give good results anyway. However, in some cases, they may neglect to find some optimizing points, or may fail to approximate an actual optimizing point well. For software to claim that it obtains rigorous bounds on all optimizing points, it would typically need to bound the range of φ over a small box $\hat{x}$ that has been proven (say, with an interval Newton method) to contain feasible points.¹⁹ This fact can lead to additional computation, including significantly more branching, in algorithms that claim to rigorously bound all optimizing points.

One aspect of rigorous branch and bound methods that can cause them not to finish in a practical amount of time, while nonrigorous ones will, is when the optimizing points are not isolated. For example, if Example 9.7 is computed using branch and bound software that does not have a facility to look for affine sets (or, more generally, manifolds) of optimizing points, then the entire set of optimizers must be covered by small boxes, and the branch and bound algorithm will produce, through branching, numerous small boxes near the set that covers the optimizing points. This can be impractical, especially if the number of variables is large.

9.7 Exercises

1. Fill in the details of the solution process for the golden mean

$$\alpha = (\sqrt{5} - 1)/2$$

in Remark 9.2 on page 491.

2. Fill in the details of the proof of Lemma 9.1 (on page 493).
3. Show that B_{k+1} defined in (9.12) on page 496 satisfies the quasi-Newton equation. (That is, $B_{k+1}s = y$.)
4. On graph paper, draw each triangle produced in the computations for the simplex method of Nelder and Mead (Algorithm 9.1 on page 497).

¹⁹This technique is presented in [45] and [44, §5.2.4].

Refer to the computations in Example 9.3 on page 500. Do your plots give you insight into how the Nelder–Mead method works?

5. Apply an iteration or two of the steepest descent method to the objective function in Example 9.3. To determine the λ_k , use the quadratic interpolation procedure described on page 494.
6. Repeat Problem 5, but alter the objective function to $\varphi(x) = (x/100)^2 + y^2 - 4$. What do you observe? (Note that the new problem is essentially a scaling of the old problem, and has the same solution.)
7. Consider finding the minimum of $f(x, y) = x^2 - 4x + xy - y + 6$ on the square $D = \{(x, y) \in \mathbb{R}^2 : 0 \leq x, y \leq 3\}$. Apply one iteration of the method of steepest descent with $x^{(0)} = 2, y^{(0)} = 1$, to find the point $(x^{(1)}, y^{(1)}) \in D$ that minimizes $f(x, y)$ in the descent direction.
8. Consider the following descent method for finding the minimum of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^1$ where $f \in C^1(\mathbb{R}^n)$. For $k = 0, 1, 2, \dots$

$$(a) \quad \varphi_k(t) = f(x_k - t\nabla f(x_k)).$$

$$(b) \quad t_k \in \mathbb{R} \text{ is chosen so that } \varphi_k(t_k) = \min_{t \in \mathbb{R}} \varphi_k(t).$$

$$(c) \quad x_{k+1} = x_k - t_k \nabla f(x_k).$$

Prove that $f(x_{k+1}) \leq f(x_k)$ and $f(x_{k+1}) = f(x_k)$ if and only if $\nabla f(x_k) = 0$.

In Exercises 9 to 11, consider the minimax problem

$$\min_{x \in \mathbf{x}} \left\{ \max_{1 \leq i \leq m} |f_i(x)| \right\}, \quad (9.22)$$

where $\mathbf{x}$ is an interval vector. (That is, we are minimizing the objective subject to bound constraints.) This is an example of a *nonsmooth* optimization problem, in which the gradient of the objective function does not exist. Such problems are common in various applications. Nonsmooth problems can be transformed to smooth constrained problems using *Lemaréchal's technique*: We introduce a new variable v , and transform (9.22) to

$$\begin{aligned} & \min v \\ & \text{subject to } v \geq f_i(x), \quad 1 \leq i \leq m, \\ & \quad v \geq -f_i(x), \quad 1 \leq i \leq m, \\ & \quad x \in \mathbf{x}. \end{aligned} \quad (9.23)$$

9. Suppose we wish to find the best fit of the form $g(t) = x_1 t + x_2$ in the ℓ_∞ norm to the data

t_i	0	1	2	3
y_i	1	4	5	8

for $-10 \leq x_1 \leq 10$, $-10 \leq x_2 \leq 10$. Thus, $f_i(x) = x_1 t_i + x_2 - y_i$.

- (a) Write down the transformed smooth problem using Lemaréchal's technique.
 - (b) Put the resulting linear program into standard form. (Note: You may put it into the standard form as in Definition 9.2 on page 504 or else into the form used for input to a linear programming solver, such as `linprog` from MATLAB's optimization toolbox.)
 - (c) Solve the resulting linear program. (You may either do this by hand, or use your favorite linear programming solver. In either case, however, explain clearly what you have done.)
10. Prove the equivalence of Problem (9.23) to Problem (9.22). That is, prove
- (a) if x^* solves (9.22), then x^* solves (9.23), and
 - (b) if x^* solves (9.23), then x^* solves (9.22).
11. In addition to minimax problems, Lemaréchal's technique can also be used to transform ℓ_1 problems. In particular, if the problem is

$$\min_{x \in \mathbf{x}} \left\{ \sum_{i=1}^m |f_i(x)| \right\}, \quad (9.24)$$

then the problem can be transformed by introducing variables v_i , $1 \leq i \leq m$.

- (a) Use Lemaréchal's technique to reformulate (9.24) as a smooth constrained optimization problem.
 - (b) Redo Problem 9, but solving (9.24) instead of (9.22).
 - (c) How does the ℓ_1 solution compare to the ℓ_∞ solution?
12. Consider Example 9.7 on page 512.
- (a) Convert the problem to standard form. *Hint: You should consider the upper bounds, e.g. $100A \leq 100000$ as regular inequality constraints.*
 - (b) In standard form, what is m_1 ?
 - (c) Write down the values of all of the variables in standard form (including the ones that are equal to zero) at the end points $t = 0$ and $t = 1250/0.9285$ of the parametrized set of solutions of optimizing points. How many of these variables are nonzero?
 - (d) How many variables are nonzero at the interior points of the parametrized solution set?

- (e) How does this relate to the Fundamental Theorem of Linear Programming?
13. Show that the linear program in Example 9.9 on page 514 is infeasible by drawing the solution set to each of the inequalities defining the feasible region.
14. Use the Simplex method to solve the linear program:

$$\begin{array}{ll} \text{maximize:} & 2x_1 + x_2 = z \\ \text{subject to:} & x_1 - 2x_2 \leq 4, \\ & x_1 + x_2 \leq 15, \\ & -x_1 + x_2 \leq 6, \\ & x_1 \leq 8, \\ & x_1 \geq 0, \\ & x_2 \geq 0. \end{array}$$

- (Notice that you need to introduce 4 slack variables.)
15. Convert the objective in Problem 14 to a minimization problem by defining $\varphi(x) = -2x_1 - x_2$. Next, write down the Kuhn–Tucker conditions corresponding to this problem.
16. Set up an algebraic recursion for Example 9.10 (on page 515) analogous to the recursion explained in Example 9.11. Solve your recursion to verify the results given with Example 9.10.

Chapter 10

Boundary-Value Problems and Integral Equations

In this final chapter, we discuss the numerical approximation of boundary value problems and integral equations. These often arise in real-world applications such as in population dynamics, computational mechanics, and in many other applications.

10.1 Boundary-Value Problems

In this section, we consider the linear boundary-value problem (BVP)

$$\begin{cases} y''(x) + p(x)y'(x) + q(x)y(x) = \varphi(x), & 0 < x < 1, \\ y(0) = \alpha, \quad y(1) = \beta. \end{cases} \quad (10.1)$$

Equation (10.1) is the one-dimensional *Dirichlet problem*¹ for the stationary diffusion equation. We study here three classes of numerical methods for solution of (10.1):

- (a) the shooting method,
- (b) finite-difference methods, and
- (c) Galerkin methods.

Several references for the material presented in this section are [9], [33], [39], [56], and [83].

Consider first an existence-uniqueness result for (10.1).

THEOREM 10.1

Suppose that Equation (10.1) can be put into the form

$$\begin{cases} -(r(x)y')' + s(x)y = g(x), & 0 < x < 1, \\ y(0) = \alpha, y(1) = \beta, \end{cases} \quad (10.2)$$

¹A Dirichlet problem is a problem in which values of the function (as opposed to values of derivatives) are specified on the boundary of the region for the differential equation.

where $r(x) \geq c_1 > 0$, $s(x) \geq 0$, for $0 < x < 1$ and $s, g \in C^m[0, 1]$, $r \in C^{m+1}[0, 1]$. Then there is a unique solution $y \in C^{m+2}[0, 1]$.

PROOF See [56, pp. 92–93]. □

REMARK 10.1 If $q(x) \leq 0$ on $[0, 1]$ and $p, q, \varphi \in C[0, 1]$, then the BVP (10.1) can be put into the form (10.2) with $r(x) \geq 1 > 0$, $s(x) \geq 0$, $s, g \in C[0, 1]$, $r \in C^1[0, 1]$ and hence $y \in C^2[0, 1]$. In particular, set

$$\frac{r'}{r} = p, \text{ so } r(x) = \exp\left(\int_0^x p(t)dt\right), \quad s = -qr, \text{ and } g = -\varphi r.$$

□

REMARK 10.2 The more general problem

$$\begin{cases} z''(t) + \tilde{p}(t)z'(t) + \tilde{q}(t)z(t) = \tilde{f}(t), & a < t < b, \\ z(a) = \alpha, z(b) = \beta, \end{cases} \quad (10.3)$$

can be converted to form (10.1) by setting $y(x) = z(a + (b - a)x)$, $0 < x < 1$, i.e., by setting $t = a + (b - a)x$. □

10.1.1 Shooting Method for Numerical Solution of (10.1)

In the *shooting method*, we solve a boundary value problem by iteratively solving associated initial value problems. Consider

$$\begin{cases} (a) & y''(x) + p(x)y'(x) + q(x)y(x) = f(x), \\ (b) & y(0) = \alpha, y(1) = \beta, \end{cases} \quad (10.4)$$

and the associated initial-value problem

$$\begin{cases} (a) & y''(x) + p(x)y'(x) + q(x)y(x) = f(x), \\ (b) & y(0) = \alpha, y'(0) = \gamma, \end{cases} \quad (10.5)$$

for some unknown γ . The theory of solutions to the IVP (10.5) is well-known. For example, if p, q , and f are continuous on $[0, 1]$, then existence of a unique solution of (10.5) is assured. Denote the solution of (10.5) by $Y(x, \gamma)$, and recall that every solution of (10.5a) is a linear combination of two particular solutions $Y^{(1)}(x)$ and $Y^{(2)}(x)$ which satisfy, say

$$\begin{cases} (a) & Y^{(1)}(0) = \alpha, Y^{(1)'}(0) = 0, \\ (b) & Y^{(2)}(0) = \alpha, Y^{(2)'}(0) = 1. \end{cases} \quad (10.6)$$

Then the unique solution of (10.5a) which satisfies (10.5b) is

$$Y(x, \gamma) = (1 - \gamma)Y^{(1)}(x) + \gamma Y^{(2)}(x). \quad (10.7)$$

Now, if we take γ such that

$$Y(1, \gamma) = (1 - \gamma)Y^{(1)}(1) + \gamma Y^{(2)}(1) = \beta, \quad (10.8)$$

then $y(x) = Y(x, \gamma)$ is a solution of BVP (10.4).

REMARK 10.3 There is at most one root of Equation (10.8), that is,

$$\gamma = \frac{\beta - Y^{(1)}(1)}{Y^{(2)}(1) - Y^{(1)}(1)},$$

provided $Y^{(2)}(1) - Y^{(1)}(1) \neq 0$. If $Y^{(2)}(1) - Y^{(1)}(1) = 0$, there may be no solution to BVP (10.4). A solution exists in this case provided $\beta = Y^{(1)}(1) = Y^{(2)}(1)$, but is not unique since, by (10.7), $Y(x, \gamma)$ is a solution for arbitrary γ . $\square$

Example 10.1

Consider three boundary-value problems. In the first problem, there exists a unique solution; in the next problem, there is no solution; in the final problem, there are an infinite number of solutions.

$$(A) \quad \begin{cases} y'' + (\frac{\pi}{2})^2 y = 0, & 0 < x < 1, \\ y(0) = 1, & y(1) = 1. \end{cases}$$

For this problem,

$$\begin{aligned} y^{(1)}(x) &= \cos \frac{\pi}{2}x, \\ y^{(2)}(x) &= \cos \frac{\pi}{2}x + \frac{2}{\pi} \sin \frac{\pi}{2}x. \end{aligned}$$

The unique solution is $y(x) = \cos \frac{\pi}{2}x + \sin \frac{\pi}{2}x$. Notice that $y^{(2)}(1) - y^{(1)}(1) \neq 0$, $\gamma = \frac{\pi}{2}$.

$$(B) \quad \begin{cases} y'' + (\pi)^2 y = 0, & 0 < x < 1, \\ y(0) = 1, & y(1) = 0. \end{cases}$$

For this problem,

$$\begin{aligned} y^{(1)}(x) &= \cos \pi x, \\ y^{(2)}(x) &= \cos \pi x + \frac{1}{\pi} \sin \pi x. \end{aligned}$$

Notice that $\beta \neq y^{(1)}(1)$ and $y^{(2)}(1) - y^{(1)}(1) = 0$. There are no constants A and B such that $y(x) = A \cos \pi x + B \sin \pi x$ satisfies $y(0) = 1$, $y(1) = 0$.

$$(C) \quad \begin{cases} y'' + (\pi)^2 y = 0, & 0 < x < 1, \\ y(0) = 1, & y(1) = -1. \end{cases}$$

Here,

$$y^{(1)}(x) = \cos \pi x, \quad y^{(2)}(x) = \cos \pi x + \frac{1}{\pi} \sin \pi x.$$

Notice that $y^{(2)}(1) - y^{(1)}(1) = 0$, but $\beta = -1 = y^{(1)}(1)$. The solutions are $y(x) = \cos \pi x + B \sin \pi x$ for any number B .

□

We assume in the following that there is a unique solution to the BVP (10.4). The procedure described by (10.5)–(10.8) is therefore valid since $y^{(2)}(1)$ cannot equal $y^{(1)}(1)$.

The previous discussion motivates the shooting method for numerical solution of BVP (10.4). If we can find γ such that the solution of IVP (10.5) when evaluated at $x = 1$ is equal to β , then we have solved the BVP (10.4). This leads to the following procedure for the shooting method.

(a) Replace (10.5) by the system:

$$\begin{cases} w_1'(x) = w_2(x), \\ w_2'(x) = f(x) - p(x)w_2(x) - q(x)w_1(x), & 0 < x < 1, \\ w_1(0) = \alpha, & w_2(0) = z, \end{cases} \quad (10.9)$$

with $w_1(x) = y(x)$ and $w_2(x) = y'(x)$.

(b) Solve (10.9) numerically, using, for example, a Runge–Kutta, multistep, an extrapolation method, or a software package written by experts.² Find an approximation to $w_1(1)$, say $w_1(1; z)$.

(c) If $|w_1(1; z) - \beta| < \epsilon$ then stop. Otherwise, modify z and return to Step (b).

This procedure corresponds to numerically solving the nonlinear problem $F(z) = w_1(1; z) - \beta$ with solution $F(\gamma) = 0$. Therefore, a reasonable way to update z is using the secant method, i.e.,

$$z^{(i+1)} = z^{(i)} - F(z^{(i)})(z^{(i)} - z^{(i+1)}) / (F(z^{(i)}) - F(z^{(i-1)}))$$

for $i = 1, 2, \dots$, where two starting values $z^{(0)}$ and $z^{(1)}$ are required³. Geometrically, the procedure looks as though we are “shooting” and adjusting the angle of the gun; see Figure 10.1. There, we are “shooting” over to hit the point $(1, 0)$. We adjust z until $w_1(1) \approx 0$.

²such as can be found in NETLIB

³Also, software packages for nonlinear equations can be used. Newton’s method has sometimes even been used for such problems, obtaining $F'(z)$ by differentiating the entire solution process for (10.9) using automatic differentiation techniques.

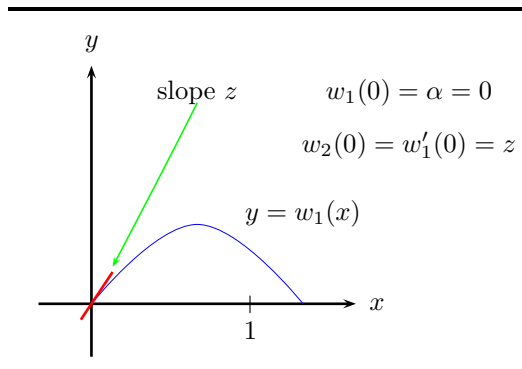


FIGURE 10.1: Illustration of the shooting method.

Nonlinear and Linear Systems The shooting method is an effective computational procedure even when the BVP (10.1) (or BVP (10.4)) is replaced by a nonlinear boundary-value problem $y''(x) = g(x, y, y')$. However, for the linear boundary-value problem (10.4), linearity can be exploited to dramatically reduce the number of computations in the shooting method. Recall the IVP (10.5). Suppose we numerically solve (10.5) twice, with the two differential initial conditions given by (10.6a) and (10.6b) to obtain approximate solutions $y_1(x)$ and $y_2(x)$ that satisfy

$$\begin{cases} y_1(0) = \alpha, & y_2(0) = \alpha, \\ y_1'(0) = 0, & y_2'(0) = 1, \end{cases} \quad (10.10)$$

We now form a linear combination of $y_1(x)$ and $y_2(x)$:

$$\hat{y}(x) = \lambda y_1(x) + (1 - \lambda) y_2(x) \quad (10.11)$$

such that $\hat{y}(1) = \beta = \lambda y_1(1) + (1 - \lambda) y_2(1)$. Thus,

$$\lambda = (\beta - y_2(1)) / (y_1(1) - y_2(1)),$$

and $\hat{y}(x)$ is an approximate solution of BVP (10.4). Hence, for the linear BVP (10.4), we solve two initial-value problems numerically, forming their sum in an appropriate manner to find an approximation to the solution $y(x)$. Accuracy of the procedure is determined by the accuracy of the numerical methods used to solve the initial-value problems.

REMARK 10.4 The shooting method can suffer from instabilities for certain problems. See [33]. $\square$

10.1.2 Finite-Difference Methods

In this section, we write the BVP (10.1) in the form

$$\begin{aligned} L\{y\} &= y'' - p(x)y' - q(x)y = r(x), \quad a < x < b, \\ y(a) &= \alpha, \quad y(b) = \beta, \end{aligned} \quad (10.12)$$

and impose the restriction $q(x) \geq c_2 > 0$, $a \leq x \leq b$. We assume in this section that $p, q, r \in C^2[a, b]$, so there exists a unique solution $y \in C^4[a, b]$, by Theorem 10.1. We divide $[a, b]$ into a uniform mesh, setting $x_i = a + ih$ for $i = 0, 1, \dots, N+1$ where $h = (b-a)/(N+1)$. Recall that

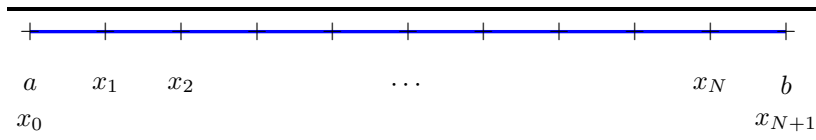


FIGURE 10.2: A finite difference mesh

$$\begin{aligned} y''(x_i) &= \frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1}))}{h^2} + \mathcal{O}(h^2), \\ y'(x_i) &= \frac{y(x_{i+1}) - y(x_{i-1}))}{2h} + \mathcal{O}(h^2). \end{aligned}$$

Thus, on the mesh,

$$\begin{cases} \frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1}))}{h^2} - p(x_i) \frac{y(x_{i+1}) - y(x_{i-1}))}{2h} \\ \quad - q(x_i)y(x_i) \approx r(x_i), \\ y(x_0) = \alpha, \quad y(x_{N+1}) = \beta, \end{cases} \quad (10.13)$$

for $i = 1, 2, \dots, N$. Equation (10.13) suggests the following numerical method for solution of (10.12). Let u_i , $0 \leq i \leq N+1$, approximate $y(x_i)$ that satisfies (10.13) exactly, i.e.,

$$\begin{cases} \text{(a)} \quad L_h\{u_i\} = \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} - p(x_i) \frac{u_{i+1} - u_{i-1}}{2h} \\ \quad \quad \quad - q(x_i)u_i = r(x_i), \\ \text{(b)} \quad u_0 = \alpha, \quad u_{N+1} = \beta, \end{cases} \quad (10.14)$$

for $i = 1, 2, \dots, N$. Multiplying (10.14a) by $-\frac{h^2}{2}$ we obtain

$$-\frac{h^2}{2}L_h\{u_i\} = -b_i u_{i-1} + a_i u_i - c_i u_{i+1} = -\frac{h^2}{2}r(x_i), \quad (10.15)$$

where

$$a_i = 1 + \frac{h^2}{2}q(x_i), \quad b_i = \frac{1}{2} \left[1 + \frac{h}{2}p(x_i) \right], \quad \text{and} \quad c_i = \frac{1}{2} \left[1 - \frac{h}{2}p(x_i) \right].$$

We can write (10.15) with boundary conditions (10.14b) as a linear system

$$Au = r, \quad \text{where} \tag{10.16}$$

$$u = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{pmatrix}, \quad r = -\frac{h^2}{2} \begin{pmatrix} r(x_1) \\ r(x_2) \\ \vdots \\ r(x_N) \end{pmatrix} + \begin{pmatrix} b_1\alpha \\ 0 \\ \vdots \\ c_N\beta \end{pmatrix} \tag{10.17}$$

and

$$A = \begin{pmatrix} a_1 & -c_1 & 0 & \cdots & & 0 \\ -b_2 & a_2 & -c_2 & & & \\ 0 & -b_3 & a_3 & -c_3 & & \\ \vdots & & \ddots & \ddots & \ddots & \\ & & & -b_{N-1} & a_{N-1} & -c_{N-1} \\ 0 & & \cdots & 0 & -b_N & a_N \end{pmatrix}.$$

To find $\{u_i\}_{i=1}^N$, we solve linear system (10.16) with tridiagonal coefficient matrix A . If we require the mesh spacing h to be small enough to ensure

$$\frac{h}{2}|p(x_i)| \leq 1 \quad \text{for } i = 1, 2, \dots, N. \tag{10.18}$$

then, since $q(x) > 0$, $|a_i| = a_i$, $|b_i| = b_i$, and $|c_i| = c_i$. Furthermore, since

$$1 + \frac{h^2}{2}q(x_i) > \frac{1}{2} \left[1 + \frac{h}{2}p(x_i) \right] + \frac{1}{2} \left[1 - \frac{h}{2}p(x_i) \right] = 1$$

for each i ,

$$|a_i| > |b_i| + |c_i| \quad \text{for } i = 1, 2, \dots, N.$$

Hence, A is strictly diagonally dominant,⁴ and the system (10.16) can be solved using direct factorization methods, which are very efficient for tridiagonal systems.

We now consider the error in approximating $\{y(x_i)\}_{i=1}^N$ by $\{u_i\}_{i=1}^N$.

⁴and thus, nonsingular

DEFINITION 10.1 We denote the local truncation error τ_i by

$$\tau_i = L_h\{y(x_i)\} - L\{y(x_i)\}.$$

Assuming $y \in C^4[a, b]$,

$$\begin{aligned}\tau_i &= \left[\frac{y(x_i + h) - 2y(x_i) + y(x_i - h)}{h^2} - y''(x_i) \right] \\ &\quad - p(x_i) \left[\frac{y(x_i + h) - y(x_i - h)}{2h} - y'(x_i) \right] \\ &= \frac{h^2}{12} \left[y^{(4)}(\xi_i) - 2p(x_i)y^{(3)}(\zeta_i) \right],\end{aligned}$$

where $\xi_i, \zeta_i \in [x_{i-1}, x_{i+1}]$.

We now have the following error estimates:

THEOREM 10.2

If the interval width h satisfies (10.18), then

$$|u_i - y(x_i)| \leq h^2 \left(\frac{M_4 + 2P^*M_3}{12Q_3} \right), \quad i = 0, 1, 2, \dots, N+1, \quad (10.19)$$

where $y(x)$ is the solution to (10.12), $\{u_i\}_{i=0}^{N+1}$ is the solution to (10.14), and

$$\begin{cases} P^* = \max_{a \leq x \leq b} |p(x)|, & M_3 = \max_{a \leq x \leq b} |y^{(3)}(x)|, \\ M_4 = \max_{a \leq x \leq b} |y^{(4)}(x)|, & Q_* = \min_{a \leq x \leq b} |q(x)|. \end{cases} \quad (10.20)$$

PROOF Define $e_i = u_i - y(x_i)$, $i = 0, 1, 2, \dots, N+1$. Consider

$$\begin{aligned}0 &= L_h\{u_i\} - L\{y(x_i)\} \\ &= L_h\{u_i\} - L_h\{y(x_i)\} + L_h\{y(x_i)\} - L\{y(x_i)\} \\ &= L_h\{u_i\} - L_h\{y(x_i)\} + \tau_i \\ &= \frac{e_{i+1} - 2e_i + e_{i-1}}{h^2} - p(x_i) \frac{e_{i+1} - e_{i-1}}{2h} - q(x_i)e_i + \tau_i.\end{aligned}$$

Multiplying the above by $\frac{h^2}{2}$ and rearranging,

$$a_ie_i = b_ie_{i-1} + c_ie_{i+1} + \frac{h^2}{2}\tau_i, \quad i = 1, 2, \dots, N,$$

Now let

$$e = \max_{0 \leq i \leq N+1} |e_i|, \quad \tau = \max_{1 \leq i \leq N} |\tau_i|.$$

Then

$$|a_i||e_i| \leq (|b_i| + |c_i|)e + \frac{h^2}{2}\tau = e + \frac{h^2}{2}\tau,$$

since $|b_i| + |c_i| = 1$. But $|a_i| = a_i \geq 1 + \frac{h^2}{2}Q_*$ so the preceding implies that

$$(1 + \frac{h^2}{2}Q_*)|e_i| \leq e + \frac{h^2}{2}\tau \quad i = 1, 2, \dots, N.$$

Also, for $i = 0$ and $i = N + 1$, $e_0 = e_{N+1} = 0$, so the preceding also holds for $i = 0$ and $i = N + 1$. Thus, replacing $|e_i|$ by e on the left-hand side of the inequality, we obtain

$$\frac{h^2}{2}Q_*e \leq \frac{h^2}{2}\tau.$$

Hence,

$$e \leq \frac{\tau}{Q_*} \leq \frac{1}{Q_*} \frac{h^2}{12} [M_4 + 2P^*M_3].$$

□

Derivative Values as Boundary Conditions Let's consider how other boundary conditions can be handled. First, suppose that $y'(0) = \alpha$ and $y'(1) = \beta$ in place of $y(0) = \alpha$ and $y(1) = \beta$. In this case, u_0 and u_{N+1} are additional unknowns, and the system has $N + 2$ unknowns rather than N unknowns. Consider

$$y''(x) \approx \frac{y(x + \Delta x) - 2y(x) + y(x - \Delta x)}{(\Delta x)^2},$$

which is approximated by

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}, \quad i = 0, 1, \dots, N + 1.$$

At $i = 0$,

$$y''(0) \approx \frac{u_1 - 2u_0 + u_{-1}}{h^2},$$

and at $i = N + 1$,

$$y''(1) \approx \frac{u_{N+2} - 2u_{N+1} + u_N}{h^2}.$$

Thus, we need a value for u_{-1} . But

$$y'(0) = \alpha \approx \frac{1}{2h}(u_1 - u_{-1}),$$

so we may take $u_{-1} = u_1 - 2\alpha h$. Similarly, we take $u_{N+2} = u_N + 2\beta h$. Thus,

$$y''(0) \approx \frac{2u_1 - 2u_0 - 2\alpha h}{h^2}, \quad y''(1) \approx \frac{2u_N - 2u_{N+1} + 2\beta h}{h^2}, \quad (10.21)$$

and (10.14a) now provides $N + 2$ equations, $i = 0, 1, \dots, N + 1$, for the $N + 2$ unknowns. (Unfortunately, these approximations to $y''(0)$ and $y''(1)$ are only first-order accurate except in the special case when $\alpha = \beta = 0$ when they are second-order accurate, as you will show in Exercise 4.)

Second, consider the *mixed boundary conditions*

$$\begin{cases} \eta_1 y(0) + \eta_2 y'(0) = \alpha, \\ \gamma_1 y(1) + \gamma_2 y'(1) = \beta. \end{cases} \quad (10.22)$$

Approximations analogous to those discussed previously can also be used in this case. For example, we may take

$$\eta_1 y(0) + \eta_2 y'(0) = \alpha \approx \eta_1 u_0 + \eta_2 \frac{1}{2h}(u_1 - u_{-1}),$$

to obtain

$$u_{-1} = u_1 - \frac{2h(\alpha - \eta_1 u_0)}{\eta_2}.$$

10.1.3 Galerkin Methods

In this section, we consider form (10.2) of the BVP (10.1) and write it as

$$\begin{cases} -\frac{d}{dx} \left(r(x) \frac{dy}{dx} \right) + s(x)y(x) = f(x), & 0 < x < 1, \\ y(0) = y(1) = 0. \end{cases} \quad (10.23)$$

Suppose that $0 < r_{\min} \leq r(x) \leq r_{\max}$ and $0 \leq s(x) \leq s_{\max}$ for $0 \leq x \leq 1$, for some constants $r_{\min}$, $r_{\max}$, and $s_{\max}$. By Theorem 10.1, when $f, s \in C[0, 1]$ and $r \in C^1[0, 1]$, the BVP (10.23) has a unique solution $y \in C^2[0, 1]$. Now let $\varphi \in C[0, 1]$ such that $\varphi(0) = \varphi(1) = 0$ and $\varphi'(x)$ is piecewise continuous on $[0, 1]$. Otherwise, φ is arbitrary. Thus,

$$\varphi \in S = \{u \in C[0, 1] : u(0) = u(1) = 0 \text{ and } u'(x) \text{ is piecewise continuous}\}.$$

Multiplying (10.23) by φ and integrating both sides from 0 to 1, we obtain

$$\int_0^1 -\frac{d}{dx} \left(r(x) \frac{dy}{dx} \right) \varphi(x) dx + \int_0^1 s(x)y(x)\varphi(x) dx = \int_0^1 f(x)\varphi(x) dx. \quad (10.24)$$

Integrating the first expression by parts and using $\varphi(0) = \varphi(1) = 0$, we obtain

$$\int_0^1 r(x) \frac{dy}{dx} \frac{d\varphi}{dx} dx + \int_0^1 s(x)y(x)\varphi(x) dx = \int_0^1 f(x)\varphi(x) dx \quad \text{for all } \varphi \in S. \quad (10.25)$$

Equation (10.25) is called the *weak formulation* of (10.23). It is nearly equivalent to (10.23) in the sense that if y satisfies (10.25,) $y(0) = y(1) = 0$, and

$y \in C^2[0, 1]$, then y satisfies (10.23). (This is because the steps leading to (10.25) can be reversed.)

The *Galerkin Method* finds an approximation to the weak formulation (10.25) of the form

$$Y(x) = \sum_{i=1}^M a_i \varphi_i(x),$$

where the $\{\varphi_i(x)\}_{i=1}^M$ are linearly independent functions that vanish at $x = 0$ and $x = 1$, i.e., $\varphi_i(0) = \varphi_i(1) = 0$ for $i = 1, 2, \dots, M$. Let

$$\hat{S}_M = \text{span}(\varphi_1, \varphi_2, \dots, \varphi_M) \subset S,$$

and suppose that $Y \in \hat{S}_M$ satisfies

$$\int_0^1 r(x) \frac{dY}{dx} \frac{d\hat{\varphi}}{dx} dx + \int_0^1 s(x) Y(x) \hat{\varphi}(x) dx = \int_0^1 f(x) \hat{\varphi}(x) dx \quad (10.26)$$

for all $\hat{\varphi} \in \hat{S}_M$. We now show that (10.26) defines $Y(x)$ uniquely. Letting $\hat{\varphi} = \varphi_k$ for $k = 1, 2, \dots, M$ gives

$$\int_0^1 r(x) \frac{dY}{dx} \frac{d\varphi_k}{dx} dx + \int_0^1 s(x) Y(x) \varphi_k(x) dx = \int_0^1 f(x) \varphi_k(x) dx \quad (10.27)$$

for $k = 1, 2, \dots, M$. But $Y(x) = \sum_{i=1}^M a_i \varphi_i(x)$, so

$$\begin{aligned} \sum_{i=1}^M a_i \left[\int_0^1 r(x) \varphi_i'(x) \varphi_k'(x) dx + \int_0^1 s(x) \varphi_i(x) \varphi_k(x) dx \right] \\ = \int_0^1 f(x) \varphi_k(x) dx \end{aligned} \quad (10.28)$$

for $k = 1, 2, \dots, M$. We thus have the linear system

$$Aa = b \quad (10.29)$$

with

$$\begin{aligned} (A)_{ki} &= \int_0^1 r(x) \varphi_i'(x) \varphi_k'(x) dx + \int_0^1 s(x) \varphi_i(x) \varphi_k(x) dx, \\ (b)_k &= \int_0^1 f(x) \varphi_k(x) dx, \\ (a)_i &= a_i. \end{aligned}$$

REMARK 10.5 A is symmetric positive definite and hence nonsingular, so we have a unique solution a and $Y(x)$ is uniquely determined. Clearly, A

is symmetric. To show that A is positive definite, consider

$$\begin{aligned}
 v^T A v &= \sum_{i,k} \left(v_i \int_0^1 r(x) \varphi'_i(x) \varphi'_k(x) dx v_k + v_i \int_0^1 s(x) \varphi_i(x) \varphi_k(x) dx v_k \right) \\
 &= \int_0^1 r(x) \sum_i v_i \varphi'_i(x) \sum_k v_k \varphi'_k(x) dx \\
 &\quad + \int_0^1 s(x) \sum_i v_i \varphi_i(x) \sum_k v_k \varphi_k(x) dx \\
 &= \int_0^1 r(x) (v'(x))^2 dx + \int_0^1 s(x) (v(x))^2 dx \\
 &> 0 \text{ if } v \neq 0.
 \end{aligned}$$

Since $r(x) > 0$ and $s(x) \geq 0$, where $v(x) = \sum_{i=1}^M v_i \varphi_i(x)$. (Note that

$$v'(x) = \left(\sum_{i=1}^M v_i \varphi_i(x) \right)' = 0$$

only if $\sum_{i=1}^M v_i \varphi_i(x) = c$, which implies $c = 0$, since $\varphi_i(0) = \varphi_i(1) = 0$. Then, by linear independence, $v_i = 0$ for $i = 1, \dots, M$. $\square$

Before continuing, let's consider an example.

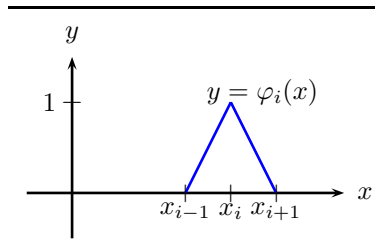
Example 10.2

Let $\hat{S}_M$ be the set of continuous piecewise linear functions defined on the partition $x_0 = 0, x_1 = h, \dots, x_M = 1$, where $h = \frac{1}{M}$, $x_i = ih$, $i = 0, 1, \dots, M$. Note that

$$\hat{S}_M = \text{span}\{\varphi_1(x), \varphi_2(x), \dots, \varphi_M(x)\},$$

where

$$\varphi_i(x) = \begin{cases} \frac{x - x_{i-1}}{h}, & x_{i-1} \leq x \leq x_i, \\ \frac{x_{i+1} - x}{h}, & x_i \leq x \leq x_{i+1}. \end{cases}$$



In this case, A is tridiagonal as well as positive definite. If $p(x) = 1 + x$, $q(x) = 0$, $f(x) = 100$, then the solution is

$$Y(x) = -100x + 100 \log(1 + x) / \log 2.$$

Solving (10.29) for a and obtaining $Y(x)$, the following L^2 -errors were obtained for various values of M .

M	$\ y - Y\ _2 = \left(\int_0^1 (y(x) - Y(x))^2 dx \right)^{\frac{1}{2}}$
2	1.780
4	0.464
8	0.118
16	0.029

Notice that error appears to be proportional to $\frac{1}{M^2} = h^2$. □

The basis functions in Example 10.2 are commonly called “hat functions” or “chapeau functions.”

REMARK 10.6 If the basis functions have small support, i.e., are nonzero on small regions, as in the above example, then the numerical method is called a *finite element method*. The matrix A is then sparse, in the above example, tridiagonal. The finite element method has become a popular and important method for solving partial differential equations. However, we are not restricted to using such basis functions. We could choose, for example, trigonometric functions, such as

$$\hat{S}_M = \text{span}(\sin \pi x, \sin 2\pi x, \dots, \sin M\pi x).$$

We could also choose polynomials, and a reasonable choice of $\hat{S}_M$ could then be

$$\hat{S}_M = \text{span} \{ x(1-x), x^2(1-x), \dots, x^{M+1}(1-x) \}.$$

(Recall that $\varphi_k(0) = \varphi_k(1) = 0$ for each k .) □

We now obtain a bound on the error. We introduce an “energy” functional. (This is related to the energy of a physical system for certain problems.) Let

$$F(u) = \int_0^1 \left(\frac{1}{2} r(x) (u'(x))^2 + \frac{1}{2} s(x) u^2(x) - f(x) u(x) \right) dx. \quad (10.30)$$

Let y be the solution of (10.23) and w be an element of S . Let $e = w - y$. Then

$$\begin{aligned} F(w) &= F(e + y) \\ &= \int_0^1 \frac{1}{2} r(x) ((e'(x))^2 \\ &\quad + 2e'(x)y'(x) + (y'(x))^2) \\ &\quad + \frac{1}{2} s(x) (e^2(x) + 2e(x)y'(x) + y^2(x)) \\ &\quad - f(x) (e(x) + y(x)) dx. \end{aligned}$$

since y satisfies the weak formulation, several terms sum to zero, and we obtain

$$F(w) = F(y) + \|e\|_F^2, \quad (10.31)$$

where $\|\cdot\|_F$ is the “energy” norm defined by

$$\|e\|_F^2 = \int_0^1 \left[\frac{1}{2} r(x) (e'(x))^2 + \frac{1}{2} e^2(x) s(x) \right] dx. \quad (10.32)$$

This is a legitimate norm since $\|e\|_F^2 = 0$ only if $e = 0$. Similarly, we can show for all $W \in \hat{S}_M$ that

$$F(W) = F(Y) + \|E\|_F^2, \quad (10.33)$$

where $E = W - Y$.

Equations (10.31) and (10.33) tell us that y minimizes F over all $u \in S$ and Y minimizes F over all $U \in \hat{S}_M$. That is, $F(y) \leq F(w)$ for all $w \in S$ and $F(Y) \leq F(W)$ for all $W \in \hat{S}_M$. Hence, $F(Y) - F(y) \leq F(W) - F(y)$ for all $W \in \hat{S}_M$.

Now, letting w in (10.31) be Y or W , we obtain

$$F(W) = F(y) + \|W - y\|_F^2$$

and

$$F(Y) = F(y) + \|Y - y\|_F^2.$$

Hence,

$$\|Y - y\|_F^2 \leq \|W - y\|_F^2 \quad \text{for all } W \in \hat{S}_M. \quad (10.34)$$

Thus, Y is the *best* approximation in $\hat{S}_M$ to y in the norm $\|\cdot\|_F$, that is Y is closest to the solution y for any function in $\hat{S}_M$ in the energy norm.

REMARK 10.7 With a considerable amount of additional work, we can show that if $y \in C^2[0, 1]$, then the error in the L_2 -norm is proportional to h^2 , i.e., $\|Y - y\|_2 \leq c_2 h^2$, where $\|g\|_2 = \left(\int_0^1 g^2(x) dx \right)^{\frac{1}{2}}$. □

REMARK 10.8 Galerkin methods can alternately be described in the following manner, let $Ly = f$, $x \in \Omega$, for example, be a boundary-value problem with $By = 0$ for $x \in \partial\Omega$, where $\partial\Omega$ denotes the boundary⁵ of Ω . Let S_N be a finite-dimensional subspace of a Hilbert space S , where $y \in S$. Then $LU_N - f = e$ where $U_N = \sum_{i=1}^N c_i \varphi_i$ and $S_N = \text{span}(\varphi_1, \varphi_2, \dots, \varphi_N)$. To find U_N , we make e orthogonal to φ_k , $k = 1, 2, \dots, N$. That is, we find the c_i to make

$$\left(L \sum_{i=1}^N c_i \varphi_i, \varphi_k \right) - (f, \varphi_k) = 0 \quad \text{for } k = 1, 2, \dots, N.$$

Notice that this approach results in system (10.27). □

Let's again consider the Galerkin method but in a functional analysis setting. We define the following Hilbert spaces:

DEFINITION 10.2

$$H^1(0, 1) = \left\{ u \in L^2(0, 1) : \frac{du(x)}{dx} \in L^2(0, 1) \right\},$$

$$H_0^1(0, 1) = \left\{ u \in H^1(0, 1) : u(0) = u(1) = 0 \right\},$$

where $L^2(0, 1)$ is the set of those Lebesgue measurable functions $u(x)$ on $(0, 1)$ that satisfy $\int_0^1 u^2(x) dx < \infty$. With the above sets, we define the inner products:

(i) On $L^2(0, 1)$: $(v, w) = \int_0^1 v(x)w(x)dx$ and $\|v\|_0^2 = (v, v)$.

(ii) On $H^1(0, 1)$ or $H_0^1(0, 1)$:

$$(v, w)_1 = \int_0^1 v(x)w(x)dx + \int_0^1 v'(x)w'(x)dx$$

and

$$\|v\|_1^2 = \int_0^1 v^2(x)dx + \int_0^1 (v'(x))^2 dx = (v, v)_1.$$

REMARK 10.9 With the above inner products, it is possible to show that $H^1(0, 1)$ and $H_0^1(0, 1)$ are complete inner product spaces, i.e., Hilbert spaces. □

⁵In the boundary value problems we have been considering, $\Omega = [0, 1]$, and $\partial\Omega$ consists of the points 0 and 1.

REMARK 10.10 Another way to define these spaces is the following. Let $V^m = \{u \in C^m(0,1) : \|u\|_m^2 = \sum_{k=0}^m \int_0^1 (u^{(k)}(x))^2 dx < \infty\}$. Then define $L^2(0,1)$ = completion of V^0 with respect to norm $\|\cdot\|_0$ and $H^1(0,1)$ = completion of V^1 with respect to norm $\|\cdot\|_1$. (Hence, V^0 is dense in $L^2(0,1)$ and V^1 is dense in $H^1(0,1)$.) $\square$

On $H_0^1(0,1)$, we define the bilinear form $B(\cdot, \cdot)$ by

$$\begin{aligned} B(v, w) &= \int_0^1 r(x)v'(x)w'(x)dx + \int_0^1 s(x)v(x)w(x)dx \\ &= (rv', w') + (sv, w) \quad \text{for } v, w \in H_0^1(0,1). \end{aligned} \quad (10.35)$$

Now note that if y is the classical solution of (10.23), then

$$\begin{aligned} B(y, v) &= \int_0^1 ry'v'dx + \int_0^1 syvdx \\ &= [ry'v]_0^1 - \int_0^1 (ry')'vdx + \int_0^1 syvdx \\ &= \int_0^1 (-(ry')' + sy)vdx \\ &= \int_0^1 fvdx \\ &= (f, v). \end{aligned}$$

That is, $B(y, v)$ satisfies

$$B(v, y) = (f, v) \quad \text{for every } v \in H_0^1(0,1). \quad (10.36)$$

Now suppose that (10.36) is satisfied for every $v \in H_0^1(0,1)$. Our question is: can we find a function $y \in H_0^1(0,1)$ which is unique solution of (10.36)? If such is the case, we call y the generalized solution of (10.36) in $H_0^1(0,1)$. Furthermore, since $C_0^2[0,1] \subset H_0^1(0,1)$, if the solution to (10.36) is sufficiently smooth, then the solution will also be the classical solution. To prove existence and uniqueness of the solution y to (10.36), we will use the Lax–Milgram Lemma:

THEOREM 10.3

(Lax–Milgram Lemma) Let H be a (real) Hilbert space and let $B(\cdot, \cdot) : H \times H \rightarrow \mathbb{R}$ be a bilinear form on H which satisfies

- (a) $|B(\Phi, \Psi)| \leq c_1 \|\Phi\| \|\Psi\|$ for all $\Phi, \Psi \in H$ (boundedness),
- (b) $B(\Phi, \Phi) \geq c_2 \|\Phi\|^2$ for all $\Phi \in H$ (coerciveness),

where c_1 and c_2 are positive constants independent of $\Phi, \Psi \in H$. Let $F : H \rightarrow \mathbb{R}^1$ be a given (real-valued) bounded linear functional on H . Then there exists a unique $u \in H$ satisfying

$$B(u, v) = F(v) \quad \text{for all } v \in H. \quad (10.37)$$

Moreover, $\|u\| \leq \|F\|/c_2$, where $\|F\| = \sup_{f \in H, f \neq 0} \frac{|F(f)|}{\|f\|}$.

Thus, to prove existence and uniqueness of y that satisfies (10.36), we will show that the conditions of the Lax–Milgram Lemma are satisfied. First, let $F(v) = (f, v)$. Then, F is a bounded linear functional on $H_0^1(0, 1)$, since

$$\begin{aligned} \|F\| &= \sup_{v \in H_0^1, v \neq 0} \frac{|(f, v)|}{\|v\|_1} \\ &= \frac{\|f\|_0 \|v\|_0}{\|v\|_1} \\ &\leq \frac{\|f\|_0 \|v\|_1}{\|v\|_1} = \|f\|_0. \end{aligned}$$

Now consider $B(\cdot, \cdot)$ defined in (10.35). Clearly, $B(\cdot, \cdot)$ is a bilinear form on $H_0^1 \times H_0^1$. Furthermore, for $v, w \in H_0^1$ we have

$$\begin{aligned} |B(w, v)| &\leq r_{\max} \int_0^1 |w' v'| dx + s_{\max} \int_0^1 |w v| dx \\ &\leq r_{\max} \|w'\|_0 \|v'\|_0 + s_{\max} \|w\|_0 \|v\|_0 \end{aligned}$$

by the Cauchy–Schwarz inequality. Thus, $|B(w, v)| \leq (r_{\max} + s_{\max}) \|w\|_1 \|v\|_1$, so condition (10.37a) of Theorem 10.3 is satisfied. Now, for $v \in H_0^1(0, 1)$,

$$\begin{aligned} B(v, v) &= \int_0^1 r(v')^2 dx + \int_0^1 s v^2 dx \\ &\geq r_{\min} \|v'\|_0^2 = r_{\min} \left(\frac{1}{2} \|v'\|_0^2 + \frac{1}{2} \|v'\|_0^2 \right). \end{aligned}$$

Then, since $v \in H_0^1(0, 1)$, Poincaré's inequality yields

$$B(v, v) \geq \frac{1}{2} r_{\min} \left(\|v'\|_0^2 + \frac{1}{c} \|v\|_0^2 \right)$$

for some constant c . Thus $B(v, v) \geq c_2 \|v\|_1^2$, where $c_2 = \frac{1}{2} r_{\min} \min(1, \frac{1}{c})$, and condition (10.35b) of the Lax–Milgram Lemma is satisfied. (Poincaré's inequality is presented and proven later as Theorem 10.5 on page 553.) Therefore, by the Lax–Milgram Lemma, there exists a unique $y \in H_0^1(0, 1)$ satisfying (10.36). Moreover, $\|y\|_1 \leq c \|f\|_0$ for some constant c .

We now approximate y , the solution of (10.36), by its Galerkin approximation y_h from a finite-dimensional subspace S_h of $H_0^1(0, 1)$. The approximate solution y_h is required to satisfy

$$B(y_h, v_h) = (f, v_h) \quad \text{for all } v_h \in S_h. \quad (10.38)$$

The existence and uniqueness of $y_h \in S_h$ is guaranteed by the Lax-Milgram Lemma, since S_h is also a Hilbert space. Now consider the error in the approximation. We have the following result.

THEOREM 10.4

Let y satisfy (10.36) and let y_h satisfy (10.38). Then

$$\|y - y_h\|_1 \leq \left(1 + \frac{c_1}{c_2}\right) \inf_{\chi \in S_h} \|y - \chi\|_1$$

where $c_1 = r_{\max} + s_{\max}$ and $c_2 = \frac{1}{2}r_{\min} \min(1, \frac{1}{c})$.

PROOF Let $\chi \in S_h$. Then

$$\|y - y_h\|_1 \leq \|y - \chi\|_1 + \|\chi - y_h\|_1. \quad (10.39)$$

In addition,

$$\begin{aligned} c_2 \|\chi - y_h\|_1^2 &\leq B(\chi - y_h, \chi - y_h) = B(\chi - y + y - y_h, \chi - y_h) \\ &= B(\chi - y, \chi - y_h) + B(y - y_h, \chi - y_h). \end{aligned} \quad (10.40)$$

Now, since $B(y_h, \Psi) = F(\Psi) = B(y, \Psi)$ for all $\Psi \in S_h$, we have

$$B(y - y_h, \Psi) = 0 \quad \text{for all } \Psi \in S_h,$$

In particular, for $\Psi = \chi - y_h$, $B(y - y_h, \chi - y_h) = 0$, so (10.40) implies

$$c_2 \|\chi - y_h\|_1^2 \leq B(\chi - y, \chi - y_h) \leq c_1 \|\chi - y\|_1 \|\chi - y_h\|_1. \quad (10.41)$$

Therefore,

$$\|\chi - y_h\|_1 \leq \frac{c_1}{c_2} \|\chi - y\|_1.$$

Then, (10.39) implies

$$\|y - y_h\|_1 \leq \left(1 + \frac{c_1}{c_2}\right) \|\chi - y\|_1$$

for any $\chi \in S_h$, i.e.,

$$\|y - y_h\|_1 \leq \left(1 + \frac{c_1}{c_2}\right) \inf_{\chi \in S_h} \|\chi - y\|_1.$$

□

COROLLARY 10.1

If the family S_h of subspaces satisfies $\lim_{h \rightarrow 0} \inf_{\chi \in S_h} \|y - \chi\|_1 = 0$, then $\lim_{h \rightarrow 0} \|y - y_h\|_1 = 0$.

Example 10.3

It can be shown that if S_h is the space of piecewise continuous linear approximations considered in Example 10.2 (the “hat function” example on page 546) and $y \in H^2(0, 1) \cap H_0^1(0, 1)$, then

$$\|y - y_h\|_1 \leq c_1 h \|y\|_2 \quad \text{and} \quad \|y - y_h\|_0 \leq c_2 h^2 \|y\|_2,$$

where

$$\|w\|_0^2 = \int_0^1 w^2(x) dx, \quad \|w\|_1^2 = \int_0^1 (w^2(x) + (w'(x))^2) dx,$$

and

$$\|w\|_2^2 = \int_0^1 [w^2(x) + (w'(x))^2 + (w''(x))^2] dx.$$

□

THEOREM 10.5

Poincaré's inequality in one-dimension: For $f \in H_0^1(0, 1)$ then $\|f\|_0 \leq \|f'\|_0$.

Proof: Since $f \in H_0^1(0, 1)$, $f(x) = \int_0^x f'(x) dx$. Using the Cauchy–Schwarz inequality in $L^2(0, 1)$,

$$\begin{aligned} |f(x)| &\leq \int_0^x |f'(x)| dx \\ &\leq \left(\int_0^x dx \right)^{\frac{1}{2}} \left(\int_0^x |f'(x)|^2 dx \right)^{\frac{1}{2}} \\ &\leq \left(\int_0^1 |f'(x)|^2 dx \right)^{\frac{1}{2}} = \|f'\|_0. \end{aligned}$$

Thus,

$$\left(\int_0^1 |f(x)|^2 dx \right)^{\frac{1}{2}} \leq \left(\int_0^1 \|f'(x)\|_0^2 dx \right)^{\frac{1}{2}} = \|f'\|_0.$$

REMARK 10.11 In practice, Corollary 10.1 indicates that by reducing the step size h , the finite element approximation in the subspace S_h can be made as close as desired to the exact solution. In other words, decreasing the

step size h (or refining the mesh) yields a better approximation. This corresponds to the h version of the finite element method [92]. Alternatively, one may enrich the subspace S_h by employing a higher-order polynomial degree approximation instead of mesh refinement. This corresponds to the p version of the finite element method [92, 81]. In the last several years, there has been a significant amount of research in employing both mesh refinement and polynomial degree refinement simultaneously to obtain the approximate finite element solution (which is called the hp -version) with numerous applications in a variety of areas [81, 82]. $\square$

10.2 Approximation of Integral Equations

Integral equations occasionally arise in applications.⁶ In this section, we consider numerical methods for solving Volterra and Fredholm integral equations of second kind. A *Volterra equation of second kind* has the form

$$f(t) - \int_0^t K(t, s, f(s))ds = g(t), \quad 0 \leq t \leq T, \quad (10.42)$$

and a *Fredholm integral equation of second kind* has the form

$$f(t) - \int_0^T K(t, s, f(s))ds = g(t), \quad 0 \leq t \leq T. \quad (10.43)$$

Several references on the numerical treatment of integral equations are [7], [20], [53], and [67].

10.2.1 Volterra Integral Equations of Second Kind

In this section, we study nonlinear Volterra equations of the second kind

$$f(t) = g(t) + \int_0^t K(t, s, f(s))ds, \quad 0 \leq t \leq T. \quad (10.44)$$

We will assume that the *kernel* $K(t, s, u)$ satisfies a Lipschitz condition with respect to the third argument.

10.2.1.1 Existence and Uniqueness of Solutions

Recall:

⁶In fact, many problems that can be posed as differential equations can also be posed as equivalent integral equations.

DEFINITION 10.3 $K(t, s, u)$ satisfies a Lipschitz condition with respect to the third argument if there is a constant $L > 0$ such that

$$|K(t, s, y) - K(t, s, z)| \leq L|y - z|$$

for $t, s \in [0, T]$, where L is independent of t, s, y , and z .

THEOREM 10.6

Assume that the functions $g(t)$ and $K(t, s, u)$ are continuous in $0 \leq s \leq t \leq T$ and $-\infty < u < \infty$ and the kernel satisfies a Lipschitz condition of the form given in Definition 10.3. Then (10.44) has a unique continuous solution for all finite T .

PROOF We define successive iterates by

$$f_n(t) = g(t) + \int_0^t K(t, s, f_{n-1}(s)) ds \quad (10.45)$$

for $n = 1, 2, 3, \dots$, with $f_0(t) = g(t)$. We subtract from (10.45) a similar equation with n replaced by $n - 1$. Then

$$f_n(t) - f_{n-1}(t) = \int_0^t \{K(t, s, f_{n-1}(s)) - K(t, s, f_{n-2}(s))\} ds. \quad (10.46)$$

Let $\varphi_n(t) = f_n(t) - f_{n-1}(t)$ with $\varphi_0(t) = g(t)$. We see that

$$f_n(t) = \sum_{i=0}^n \varphi_i(t). \quad (10.47)$$

Also, (10.46) and the Lipschitz condition give

$$|\varphi_n(t)| \leq L \int_0^t |\varphi_{n-1}(s)| ds. \quad (10.48)$$

We now show by induction that

$$\varphi_n(t) \leq \frac{G(Lt)^n}{n!}, \quad 0 \leq t \leq T, \quad n = 0, 1, \dots \quad (10.49)$$

where $G = \max_{0 \leq t \leq T} |g(t)|$. Clearly, this is true for $n = 0$. Suppose it is true for $n - 1$. Then, by (10.48),

$$|\varphi_n(t)| \leq L \int_0^t \frac{G(Ls)^{n-1}}{(n-1)!} ds \leq \frac{G(Lt)^n}{n!}.$$

This bound shows that the sequence $f_n(t)$ in (10.47) converges, and we write

$$f(t) = \sum_{i=0}^{\infty} \varphi_i(t). \quad (10.50)$$

We now show that this $f(t)$ satisfies (10.44). The series (10.50) is uniformly convergent, since the terms $\varphi_i(t)$ are dominated by $\frac{G(LT)^i}{i!}$. Hence, $f(t)$ exists and is continuous. To prove that $f(t)$ defined by (10.50) satisfies the original equation (10.44), set

$$f(t) = f_n(t) + \Delta_n(t). \quad (10.51)$$

From equation (10.45),

$$f(t) - \Delta_n(t) = g(t) + \int_0^t K(t, s, f(s) - \Delta_{n-1}(s)) ds,$$

so

$$\begin{aligned} f(t) - g(t) - \int_0^t K(t, s, f(s)) ds \\ = \Delta_n(t) + \int_0^t [K(t, s, f(s) - \Delta_{n-1}(s)) - K(t, s, f(s))] ds. \end{aligned}$$

Applying the Lipschitz condition gives

$$\left| f(t) - g(t) - \int_0^t K(t, s, f(s)) ds \right| \leq |\Delta_n(t)| + Lt \|\Delta_{n-1}\|, \quad (10.52)$$

where $\|\Delta_{n-1}\| = \max_{0 \leq s \leq t} |\Delta_{n-1}(s)|$. But $\lim_{n \rightarrow \infty} |\Delta_n(t)| = 0$, so by taking n large enough, the right member of (10.52) can be made as small as desired. It follows that the function defined by (10.50) satisfies:

$$f(t) = g(t) + \int_0^t u(t, s, f(s)) ds.$$

To show uniqueness, we assume existence of another continuous solution $\tilde{f}(t)$. Then,

$$|f(t) - \tilde{f}(t)| = \left| \int_0^t \{K(t, s, f(s)) - K(t, s, \tilde{f}(s))\} ds \right| \quad (10.53)$$

$$\leq L \int_0^t |f(s) - \tilde{f}(s)| ds \leq BLt, \quad (10.54)$$

since $|f(s) - \tilde{f}(s)|$ must be bounded by some constant B . Thus, $|f(t) - \tilde{f}(t)| \leq BLt$. By replacing $|f(s) - \tilde{f}(s)|$ by BLs in (10.54), we obtain that $|f(s) - \tilde{f}(s)|$ must be bounded by $B(Lt)^2/2$. Repeating this process n times leads us to

$$|f(t) - \tilde{f}(t)| \leq B \frac{(Lt)^n}{n!}$$

for any n . We therefore conclude that $f(t) = \tilde{f}(t)$. □

REMARK 10.12 If $\left| \frac{\partial K(t, s, u)}{\partial u} \right| < L$ for $0 \leq s, t \leq T$ and $-\infty < u < \infty$, then K satisfies a Lipschitz condition of the form given in Definition 10.3. □

10.2.1.2 Numerical Solution Techniques

We now consider numerical solution of the Volterra equation of second kind (10.44). We assume we have a numerical integration rule of the form

$$\int_0^{nh} \varphi(t) dt \approx h \sum_{i=0}^n w_{ni} \varphi(t_i), \quad (10.55)$$

where $\varphi(t)$ is any continuous integrand and w_{ni} are integration weights. (For example, if $t_i = ih$, $w_{n0} = w_{nn} = \frac{1}{2}$, and $w_{ni} = 1$ for $i = 1, 2, \dots, n-1$, Equation (10.55) is the composite trapezoidal rule.) Using (10.55) to replace the integral in (10.44), we obtain the iteration equation

$$F_n = g(t_n) + h \sum_{i=0}^n w_{ni} K(t_n, t_i, F_i) \quad \text{for } n = 1, 2, 3, \dots, \quad (10.56)$$

with $F_0 = f(0) = g(0)$, where F_n denotes the approximate value of $f(t_n)$.

REMARK 10.13 If the w_{ni} are bounded and h is sufficiently small, then F_n for $n = 1, 2, 3, \dots$ can be calculated from (10.56). That is, if we do fixed point iteration $F_n^{(k+1)} = G(F_n^{(k)})$ for $k = 0, 1, 2, \dots$ with $G : \mathbb{R} \rightarrow \mathbb{R}$ and G equal to the right member of (10.56), then the fixed point iteration will converge for h sufficiently small and the w_{ni} bounded. (You will prove this later in Exercise 9.) $\square$

We now analyze the error in this numerical method for the special case that the composite trapezoidal rule is used to approximate the integral in (10.44). The following lemma will be useful.

LEMMA 10.1

Suppose that $\epsilon_0 = 0$ and

$$\epsilon_n \leq Bh^2 + Ah \sum_{i=0}^{n-1} \epsilon_i \quad \text{for } n = 1, 2, 3, \dots, \quad (10.57)$$

where $A, B > 0$. Then,

$$\epsilon_i \leq Bh^2 e^{Aih} \quad \text{for } i = 0, 1, 2, \dots. \quad (10.58)$$

PROOF Inequality (10.58) is clearly true for $n = 0$. Suppose that it is true for $i = 0, 1, 2, \dots, n-1$. We will show that it is true for $i = n$ and thus

true for all i . By (10.57),

$$\begin{aligned}\epsilon_n &\leq Bh^2 + Ah \sum_{i=0}^{n-1} \epsilon_i \\ &\leq Bh^2 + Ah^3 B \sum_{i=0}^{n-1} (e^{Ah})^i = Bh^2 + Ah^3 B \frac{e^{Anh} - 1}{e^{Ah} - 1} \\ &\leq Bh^2 + Ah^3 B \frac{e^{Anh} - 1}{Ah} = Bh^2 + Bh^2 e^{Anh} - Bh^2 = Bh^2 e^{Anh}.\end{aligned}$$

Inequality (10.58) is thus true for all i . (Note that $\frac{1}{e^x - 1} \leq \frac{1}{x}$ for $x > 0$.) $\square$

THEOREM 10.7

Assume that equation (10.44) has a unique continuous solution f and the kernel satisfies a Lipschitz condition of the form given in Definition 10.3. Let $\hat{K}(t, s) = K(t, s, f(s))$ and assume that $\hat{K} \in \mathcal{C}^2([0, T] \times [0, T])$. Then, assuming that $0 < hL < 1$ and the composite trapezoidal rule is used in (10.56),

$$\epsilon_n \leq \frac{M_2 h^2 T}{6} e^{2LT} \quad \text{for } n = 0, 1, 2, \dots, T/h. \quad (10.59)$$

PROOF Let $\epsilon_n = |F_n - f(t_n)|$ for $n = 0, 1, 2, \dots$. Subtracting (10.44) from (10.56) gives

$$\begin{aligned}F_n - f(t_n) &= \int_0^{t_n} K(t_n, s, f(s)) ds - h \sum_{i=0}^n w_{ni} K(t_n, t_i, F_i) \\ &= \int_0^{t_n} K(t_n, s, f(s)) ds - h \sum_{i=0}^n w_{ni} K(t_n, t_i, f(s_i)) \\ &\quad + h \sum_{j=0}^n w_{ni} [K(t_n, t_i, f(s_i)) - K(t_n, t_i, F_i)].\end{aligned}$$

Thus,

$$\begin{aligned}\epsilon_n &\leq \left| \int_0^{t_n} \hat{K}(t_n, s) ds - h \sum_{i=0}^n w_{ni} \hat{K}(t_n, s_i) \right| + hL \sum_{i=0}^n |w_{ni}| \epsilon_i \\ &\leq \max_{0 \leq s \leq t_n} \left| \frac{\partial^2 \hat{K}(t_n, s)}{\partial s^2} \right| \frac{h^2 t_n}{12} + \frac{1}{2} hL \epsilon_n + hL \sum_{i=0}^{n-1} \epsilon_i,\end{aligned}$$

where the first term is due to the error in composite trapezoidal rule. Hence, letting

$$M_2 = \max_{0 \leq s, t \leq T} \left| \frac{\partial^2 \hat{K}(t, s)}{\partial s^2} \right|$$

and using the assumption $hL < 1$,

$$\epsilon_n \leq \frac{M_2 h^2 T}{6} + 2hL \sum_{i=0}^{n-1} \epsilon_i.$$

Now, using Lemma 10.1 with $B = \frac{M_2 T}{6}$ and $A = 2L$, we have

$$\epsilon_i \leq \frac{M_2 h^2 T}{6} e^{2Lih} \quad \text{for } i = 0, 1, 2, \dots,$$

so

$$\epsilon_n \leq \frac{M_2 h^2 T}{6} e^{2LT} \quad \text{for any } 0 \leq n \leq T/h.$$

□

REMARK 10.14 Theorem 10.7 says that the order of convergence of this method is $\mathcal{O}(h^2)$. Consider the problem

$$f(t) = e^t - \int_0^t e^{(t-s)} f(s) ds,$$

with exact solution $f(t) \equiv 1$. Calculated errors using this method are given below.

Errors = $ F_i - f(t_i) $			
t	$h = 0.1$	$h = 0.05$	$h = 0.025$
0.2	1.7×10^{-4}	4.2×10^{-5}	1.0×10^{-5}
0.4	3.3×10^{-4}	8.3×10^{-5}	2.1×10^{-5}
0.6	5.0×10^{-4}	1.3×10^{-4}	3.1×10^{-5}
0.8	6.7×10^{-4}	1.7×10^{-4}	4.2×10^{-5}
1.0	8.3×10^{-4}	2.1×10^{-4}	5.2×10^{-5}

The calculated results illustrate that the error is proportional to h^2 .

□

REMARK 10.15 More information about numerical methods for Volterra integral equations can be found in [53].

□

10.2.2 Fredholm Integral Equations of Second Kind

In this section, we study linear Fredholm equations of the second kind

$$f(t) = g(t) + \int_0^1 K(t, s) f(s) ds \quad \text{for } 0 \leq t \leq 1. \quad (10.60)$$

10.2.2.1 Existence and Uniqueness

We begin with an existence and uniqueness result for the solution of (10.60).

THEOREM 10.8

Assume that $g \in \mathcal{C}[0, 1]$ and $K \in \mathcal{C}([0, 1] \times [0, 1])$. In addition, assume

$$M = \max_{0 \leq t, s \leq 1} |K(t, s)| < 1.$$

Then, Equation (10.60) has a unique continuous solution on $[0, 1]$.

PROOF We define a sequence of continuous functions on $[0, 1]$ by

$$x_0(t) = g(t) \quad x_1(t) = g(t) + \int_0^1 K(t, s)x_0(s)ds, \quad \dots,$$

that is,

$$x_n(t) = g(t) + \int_0^1 K(t, s)x_{n-1}(s)ds \quad (10.61)$$

for $n = 1, 2, 3, \dots$, where $x_0(t) = g(t)$. It can be easily seen that

$$\begin{aligned} x_n(t) = g(t) + \int_0^1 K(t, s)g(s)ds + \int_0^1 K_2(t, s)g(s)ds \\ + \dots + \int_0^1 K_n(t, s)g(s)ds, \end{aligned} \quad (10.62)$$

where

$$K_n(t, s) = \int_0^1 K(t, u)K_{n-1}(u, s)du, \quad n \geq 2, \quad (10.63)$$

with $K_1(t, s) = K(t, s)$. It follows from (10.63) that

$$K_n(t, s) = \underbrace{\int_0^1 \dots \int_0^1}_{n-1} K(t, t_1)K(t_1, t_2)K(t_2, t_3) \dots K(t_{n-1}, s)dt_1dt_2 \dots dt_{n-1}, \quad (10.64)$$

which leads to:

$$K_n(t, s) = \int_0^1 K_p(t, u)K_{n-p}(u, s)du, \quad 1 \leq p \leq n-1. \quad (10.65)$$

Formula (10.62) can also be written

$$x_n(t) = g(t) + \int_0^1 \left\{ \sum_{j=1}^n K_j(t, s) \right\} g(s)ds, \quad n \geq 1. \quad (10.66)$$

But the series $\sum_{j=1}^{\infty} K_j(t, s)$ is uniformly convergent with respect to $(t, s) \in [0, 1] \times [0, 1]$. Indeed from (10.64),

$$|K_n(t, s)| \leq M^n, \quad n \geq 1, \quad (10.67)$$

which shows that the series $\sum_{j=1}^{\infty} K_j(t, s)$ is dominated by the series with j -th term M^j , that is, a convergent geometric series. Let's denote

$$R(t, s) = \sum_{j=1}^{\infty} K_j(t, s). \quad (10.68)$$

It follows that $R(t, s)$ is a continuous function on $[0, 1] \times [0, 1]$. From (10.62) and (10.67),

$$|x_n(t) - x_{n-1}(t)| \leq NM^n, \quad (10.69)$$

where $N = \sup_{0 \leq t \leq 1} |g(t)|$. Inequality (10.69) shows that sequence $\{x_n(t)\}_{n=1}^{\infty}$ is uniformly convergent on $[0, 1]$. Taking (10.66) and (10.68) into account, we obtain

$$x(t) = g(t) + \int_0^1 R(t, s)g(s)ds, \quad (10.70)$$

where $x(t)$ is the limit of sequence $\{x_n(t)\}_{n=1}^{\infty}$. From the construction of this sequence, it follows that $x(t)$ is the solution of (10.60). Consequently, (10.70) gives a solution of (10.60).

To prove uniqueness, assume that $y(t)$ is also a solution of the same equation:

$$y(t) = g(t) + \int_0^1 K(t, s)y(s)ds. \quad (10.71)$$

Using the recurrence formula (10.61) for $x_n(t)$, we obtain

$$\begin{aligned} |y(t) - x_n(t)| &\leq \int_0^1 |K(t, s)| |y(s) - x_{n-1}(s)| ds \\ &\leq M \int_0^1 |y(s) - x_{n-1}(s)| ds. \end{aligned}$$

Consequently, denoting $\alpha_n = \sup_{0 \leq t \leq 1} |y(t) - x_n(t)|$, we obtain

$$\alpha_n \leq M\alpha_{n-1} \leq \cdots \leq M^n\alpha_0.$$

But this shows that $\alpha_n \rightarrow 0$ as $n \rightarrow \infty$, that is, $y(t) = \lim_{n \rightarrow \infty} x_n(t) = x(t)$. $\square$

10.2.2.2 Numerical Solution Techniques

We now consider numerical solution of (10.60). We assume that we have a numerical integration rule of the form

$$\int_0^1 \varphi(t) dt \approx h \sum_{i=0}^N w_i \varphi(t_i), \quad (10.72)$$

where $t_i = ih$ and $h = 1/N$, and $\sum_{i=0}^N w_i = N$, with $w_i > 0$ for $i = 0, 1, 2, \dots, N$.

Approximating the integral in equation (10.60) by the numerical integration scheme (10.72) suggests the following approximate method of solution:

$$F_i = g(t_i) + h \sum_{j=0}^N w_j K(t_i, t_j) F_j, \quad (10.73)$$

for $i = 0, 1, 2, \dots, N$, where $F_i \approx f(t_i)$ and $t_i = ih = i/N$ and $w_j > 0$ for $j = 0, 1, 2, \dots, N$.

We now assume that the quadrature rule has accuracy of order p and that K and $f(t)$ are sufficiently smooth for this order of accuracy to be attained. Specifically, it is assumed that there is a constant $c > 0$ such that

$$\max_{0 \leq x \leq 1} \left| \int_0^1 K(x, t) f(t) dt - h \sum_{j=0}^N w_j K(x, t_j) f(t_j) \right| \leq ch^p. \quad (10.74)$$

We now have the following convergence result:

THEOREM 10.9

Assume that the conditions of Theorem 10.8 hold, so equation (10.60) has a unique continuous solution. Also, assume that inequality (10.74) is valid. Then,

$$\left(\sum_{i=0}^N h w_i (F_i - f(t_i))^2 \right)^{\frac{1}{2}} \leq \frac{ch^p}{1 - M}. \quad (10.75)$$

PROOF Letting $t = t_i$ in (10.60), subtract this equation from (10.73) to obtain:

$$F_i - f(t_i) = - \int_0^1 K(t, s) f(s) ds + h \sum_{j=0}^N w_j K(t_i, t_j) F_j. \quad (10.76)$$

Letting $\epsilon_i = F_i - f(t_i)$,

$$\epsilon_i = h \sum_{j=0}^N w_j K(t_i, t_j) f(t_j) - \int_0^1 K(t, s) f(s) ds + h \sum_{j=0}^N w_j K(t_i, t_j) \epsilon_j. \quad (10.77)$$

Squaring (10.77), multiplying by hw_i , and summing from $i = 0$ to N gives

$$\sum_{i=0}^N hw_i \epsilon_i^2 = \sum_{i=0}^N hw_i \left(E_i + \sum_{j=0}^N w_j K(t_i, t_j) \epsilon_j \right)^2, \quad (10.78)$$

where

$$E_i = \sum_{j=0}^N hw_j f_j K(t_i, t_j) - \int_0^1 K(t_i, s) f(s) ds \leq ch^p.$$

Hence,

$$\begin{aligned} \sum_{i=0}^N hw_i \epsilon_i^2 &= \sum_{i=0}^N hw_i E_i^2 + 2 \sum_{i=1}^N hw_i E_i \sum_{j=0}^N hw_j K(t_i, t_j) \epsilon_j \\ &\quad + \sum_{i=0}^N hw_i \left(\sum_{j=0}^N hw_j K(t_i, t_j) \epsilon_j \right)^2. \end{aligned} \quad (10.79)$$

Letting

$$E^2 = \sum_{i=0}^N hw_i E_i^2, \quad \alpha = \left(\sum_{i=0}^N \sum_{j=0}^N hw_i hw_j K^2(t_i, t_j) \right)^{\frac{1}{2}}, \quad \epsilon^2 = \sum_{i=0}^N hw_i \epsilon_i^2$$

and applying the Cauchy-Schwarz inequality gives

$$\epsilon^2 \leq E^2 + 2\alpha E\epsilon + \alpha^2 \epsilon^2. \quad (10.80)$$

However this inequality implies that $\epsilon \leq \frac{E}{1-\alpha}$. Noticing that $E \leq ch^p$ and $\alpha \leq M$ gives the desired result. $\square$

Galerkin Approach Now consider a Galerkin approach for approximating the solution of the linear Fredholm equation of second kind (10.60). Let's write (10.60) in the form

$$Lf = g(t) \quad \text{for } t \in [0, 1], \quad (10.81)$$

where

$$Lf = f(t) - \int_0^1 K(t, s) f(s) ds.$$

Let $\mathcal{L}^2(0, 1)$ be the Hilbert space of Lebesgue measurable functions $y(t)$ such that $\int_0^1 y^2(t) dt < \infty$. Let $S_N \subset \mathcal{L}^2(0, 1)$ be a finite-dimensional subspace of $\mathcal{L}^2(0, 1)$ with $\{\varphi_i(t)\}_{i=1}^N$ a basis for S_N . One way of defining a Galerkin method is as follows.

Let $F_N \in S_N$ be an approximation to $f \in \mathcal{L}^2(0, 1)$, where $F_N(t) = \sum_{i=1}^N c_i \varphi_i(t)$ and c_i are to be determined. Then

$$LF_N - g = e \in \mathcal{L}^2(0, 1). \quad (10.82)$$

To find c_i , $i = 1, 2, \dots, N$, we make the error e orthogonal to $\varphi_1, \varphi_2, \dots, \varphi_N$. That is,

$$(LF_N - g, \varphi_j) = 0 \quad \text{for } j = 1, 2, \dots, N, \quad (10.83)$$

where $(v, u) = \int_0^1 u(x)v(x)dx$. Substituting into F_N , we obtain the following linear system to solve for the c_i 's:

$$\sum_{i=1}^N c_i (L\varphi_i, \varphi_j) = (g, \varphi_j) \quad \text{for } j = 1, 2, \dots, N. \quad (10.84)$$

To show that $F_N(t)$ is a good approximation to $f(t)$, we use the following lemma. Lemma 10.2 can also be used to show that the coefficient matrix of (10.84) is positive definite and hence nonsingular assuming that the kernel is symmetric. (See Exercise 11.)

LEMMA 10.2

Suppose that the conditions of Theorem 10.8 hold so that (10.60) has a unique continuous solution. Then

$$(L\varphi, \varphi) \geq (1 - M)\|\varphi\|^2 \quad \text{for any } \varphi \in \mathcal{L}^2(0, 1). \quad (10.85)$$

PROOF

$$\begin{aligned} (L\varphi, \varphi) &= (\varphi, \varphi) - \int_0^1 \int_0^1 K(t, s)\varphi(s)\varphi(t)dsdt \\ &\geq (\varphi, \varphi) - \int_0^1 \left(\int_0^1 K^2(t, s)ds \right)^{1/2} \left(\int_0^1 \varphi^2(s)ds \right)^{1/2} |\varphi(t)|dt \\ &\geq (\varphi, \varphi) - \int_0^1 \varphi^2(t)dt \left(\int_0^1 \int_0^1 K^2(t, s)dsdt \right)^{\frac{1}{2}} \\ &\geq (\varphi, \varphi)(1 - M), \end{aligned}$$

by applying the Cauchy-Schwarz inequality to obtain the first and second inequalities in the preceding computation. $\square$

By (10.81), the solution $f \in \mathcal{L}^2(0, 1)$ satisfies

$$(Lf - g, \varphi) = 0 \quad \text{for all } \varphi \in \mathcal{L}^2(0, 1). \quad (10.86)$$

Hence, (10.83) and (10.86) give

$$(Lf - LF_N, \Phi_N) = 0 \quad \text{for all } \Phi_N \in S_N \subset \mathcal{L}^2(0, 1). \quad (10.87)$$

Combining the above results gives

$$\begin{aligned} (1 - M)\|f - F_N\|^2 &\leq (L(f - F_N), (f - F_N)) \\ &= (L(f - F_N), (f - \Phi_N)) \\ &\leq (L(f - F_N), L(f - F_N))^{\frac{1}{2}} \|f - \Phi_N\| \\ &\leq (1 + 2M + M^2)^{\frac{1}{2}} \|f - F_N\| \|f - \Phi_N\|, \end{aligned} \quad (10.88)$$

using $(L\varphi, L\varphi) \leq (1 + 2M + M^2)\|\varphi\|^2$ for $\varphi \in \mathcal{L}^2(0, 1)$. Hence,

$$\|f - F_N\| \leq \hat{c}\|f - \Phi_N\| \quad \text{for } \Phi_N \in S_N \subset \mathcal{L}^2(0, 1), \quad (10.89)$$

where $\hat{c} = (1 + 2M + M^2)^{\frac{1}{2}}/(1 - M)$.

REMARK 10.16 Inequality (10.89) says that if the finite-dimensional subspace S_N is chosen so that $\inf_{\Phi_N \in S_N} \|f - \Phi_N\| \rightarrow 0$ as $N \rightarrow \infty$, then $\lim_{N \rightarrow \infty} \|f - F_N\| = 0$. $\square$

REMARK 10.17 The Lax–Milgram Lemma can alternately be applied to yield result (10.89). $\square$

Example 10.4

$$f(t) - \int_0^1 K(t, s)f(s)ds = g(t) = 1 + t, \quad 0 \leq t \leq 1,$$

where

$$K(t, s) = \begin{cases} t - s, & 0 \leq s \leq t \leq 1 \\ 0, & s > t \end{cases}.$$

Let

$$F_N(t) = \sum_{i=1}^N c_i \varphi_i(t),$$

where the $\varphi_i(t)$ are the continuous piecewise linear hat functions. Specifically, let $N = 3$. Solving for $F_3(t)$ yields

$$F_3(t) = \varphi_1(t) + 1.6522\varphi_2(t) + 2.7354\varphi_3(t),$$

where

$$\begin{aligned}\varphi_1(t) &= \frac{\frac{1}{2} - t}{\frac{1}{2}}, & 0 \leq t \leq \frac{1}{2}, \\ \varphi_2(t) &= \begin{cases} \frac{t}{\frac{1}{2}}, & 0 \leq t \leq \frac{1}{2}, \\ \frac{1-t}{\frac{1}{2}}, & \frac{1}{2} \leq t \leq 1, \end{cases} \\ \varphi_3(t) &= \frac{t - \frac{1}{2}}{\frac{1}{2}}, & \frac{1}{2} \leq t \leq 1.\end{aligned}$$

The exact solution of this problem is $f(t) = e^t$. Values for the numerical solution are compared in the following table.

t	$f(t) = e^t$	$F_3(t)$
0	1	1
$\frac{1}{2}$	1.649	1.652
1	2.718	2.735

□

10.3 Exercises

1. Consider solving the boundary value problem

$$\frac{d^2y}{dx^2} = \frac{6y}{x^2}, \quad y(2) = 1, \quad y(4) = 8, \quad 2 \leq x \leq 4$$

using the shooting method. Transform the problem into a system of two first-order equations by introducing a variable $z = \frac{dy}{dx}$.

- (a) Assume $z(2) = 0$ and employ Euler's method with step size $h = 1$ to find $y(4)$.
- (b) Repeat part (a) with $z(2) = 3$.
- (c) Using the guesses for $z(2)$ and the corresponding results for $y(4)$ in parts (a) and (b), interpolate to find the value of $z(2)$ to obtain the desired result $y(4) = 8$.

2. Consider the two-point boundary-value problem

$$y''(x) - p(x)y'(x) - q(x)y(x) = r(x), \quad 0 < x < 1,$$

with $y(0) = y(1) = 0$. Assume that $q(x) \geq \alpha > 0$ for $0 \leq x \leq 1$. Consider the difference scheme

$$\frac{(y_{j+1} - 2y_j + y_{j-1}))}{h^2} - p(x_j) \frac{(y_{j+1} - y_{j-1}))}{2h} - q(x_j)y_j = r(x_j)$$

for $j = 1, 2, \dots, N-1$, with $y_0 = y_N = 0$, $x_j = jh$ and $h = 1/N$.

- (a) Determine the matrix A so that the above difference equations can be written as the linear system

$$A\mathbf{y} = h^2\mathbf{r}$$

with

$$\mathbf{y} = [y_1, y_2, \dots, y_{N-1}]^T \text{ and } \mathbf{r} = [r(x_1), r(x_2), \dots, r(x_{N-1})]^T.$$

- (b) Prove that if

$$\frac{h}{2} \max_{0 \leq x \leq 1} |p(x)| \leq 1,$$

then the $(N-1) \times (N-1)$ matrix A is strictly diagonally dominant.

3. Consider

$$\begin{cases} y'' - p(x)y' - q(x)y = r(x), & a < x < b, \\ y(a) = \alpha, & y(b) = \beta, \end{cases}$$

with $q(x) \geq c_2 > 0$. Consider the difference equations

$$\frac{(u_{i+1} - 2u_i + u_{i-1}))}{h^2} - p(x_i) \frac{(u_{i+1} - u_{i-1}))}{h} - q(x_i)u_i = r(x_i)$$

for $i = 1, \dots, N$, with $u_0 = \alpha$, $u_{N+1} = \beta$, where

$$x_i = a + ih, \quad h = \frac{b-a}{N+1}.$$

Assume that $y \in C^4[a, b]$. Prove that for h sufficiently small,

$$\max_i |u_i - y(x_i)| \leq h \left(\frac{hM_4 + 6M_2P^*}{12Q_*} \right), \quad \text{where } M_l = \max_{a \leq x \leq b} |y^{(l)}(x)|.$$

4. Show that the approximations given in (10.21) (on page 543) are accurate to second order when $\alpha = \beta = 0$, but are only first order otherwise.
5. Consider the boundary-value problem

$$-\frac{d^2y}{dx^2} = f(x), \quad y(0) = y(1) = 0,$$

with $f \in C[0, 1]$, so that $y \in C^2[0, 1]$ uniquely exists. Consider approximating $y(x)$ by

$$Y(x) = \sum_{i=1}^{M-1} a_i \varphi_i(x) \in \hat{S}_M,$$

where S_M is the set of continuous piecewise linear functions on a partition $x_0 = 0$, $x_1 = h$, $x_2 = 2h$, $\dots$, $x_M = 1$, and $h = \frac{1}{M}$. If the approximation is by the Galerkin method, then prove that

$$\|Y - y\|_F^2 = \frac{1}{2} \int_0^1 (y'(x) - Y'(x))^2 dx \leq ch \max_{0 \leq x \leq 1} |y''(x)|.$$

6. Apply the Galerkin method to approximate the solution $y(x)$ to the two-point boundary-value problem

$$\begin{cases} -y''(x) + \left(\frac{\pi}{2}\right)^2 y(x) = x, \\ y(0) = y(1) = 0. \end{cases}$$

Let

$$y(x) \approx Y(x) = c_1 \sin(\pi x) + c_2 \sin(2\pi x)$$

and find c_1 and c_2 . Also find the exact solution $y(x)$ and compare $Y(x)$ to $y(x)$ at $x = 0.25$, 0.5 , and 0.75 .

7. Show that

$$f(t) = 1 + \int_0^t \frac{\sin(t-s)}{1+f^2(s)} ds$$

has a unique continuous solution for all t .

8. Prove Remark 10.12 on page 556.
9. Prove the assertion in Remark 10.13 on page 557. (Hint: Use the contraction mapping theorem.)
10. Consider the Fredholm integral equation

$$f(t) = g(t) + \int_0^1 K(t, s) f(s) ds$$

for $0 \leq t \leq 1$, where $\max_{0 \leq t, s \leq 1} |K(t, s)| = \frac{1}{2}$. Suppose that

$$\left| \int_0^1 K(t_i, s) f(s) ds - h \sum_{j=0}^N w_j K(t_i, t_j) f(t_j) \right| \leq ch^p$$

for each i where $t_i = ih$. Also suppose that $h < \frac{1}{2}$, $0 \leq w_j \leq 1$ for all j , and $\sum_{j=0}^N hw_j = 1$. Let

$$F_i = g(t_i) + h \sum_{j=0}^N w_j K(t_i, t_j) F_j$$

for $i = 0, 1, \dots, N$. Prove that $\max_j |f(t_j) - F_j| \leq 2ch^p$.

11. Use Lemma 10.2 (on page 564) to show that system (10.84) (on page 564) has a positive definite coefficient matrix and hence is nonsingular. Assume that $K(t, s) = K(s, t)$, i.e. the kernel K is symmetric.

Appendix A

Solutions to Selected Exercises

Solutions from Chapter 1

1(c). Since $f \in C[a, b]$, we have

$$\min_{x \in [a, b]} f(x) \leq f(x_j) \leq \max_{x \in [a, b]}$$

for $j = 1, \dots, n$. Also, multiplying the inequality by w_j and summing over j from $1, \dots, n$, we get

$$\min_{x \in [a, b]} f(x) \leq \sum_{j=1}^n w_j f(x_j) \leq \max_{x \in [a, b]} f(x),$$

since $w_j \geq 0$ and $\sum_{j=1}^n w_j = 1$. The result then follows from the Intermediate Value Theorem.

5. See Example 1.3.

7. It is easy to verify that $a + (b + c) = 0.327$, which is not equal to $(a + b) + c = 0.326$.

9. Let $S_2 = a^T b = x_1 x_3 + x_2 x_4$. Note that

$$x_1^* x_3^* = x_1(1 + \epsilon_1) x_3(1 + \epsilon_3)(1 + \epsilon_{13}),$$

where $\epsilon_{13} < \delta$ corresponds to the error due to the multiplication. Similarly,

$$x_2^* x_4^* = x_2(1 + \epsilon_2) x_4(1 + \epsilon_4)(1 + \epsilon_{24}),$$

where $\epsilon_{24} < \delta$ corresponds to the error due to the multiplication. We then have

$$S_2^* \leq S_2(1 + \delta)^4 \leq S_2 e^{4\delta}.$$

16. $|x^* - y^*| = |x(1 + r_x) - y(1 + r_y)| \geq |x - y| - (|x| + |y|)R$. Therefore, to ensure $x^* \neq y^*$ we must have $R < |x - y|/(|x| + |y|)$.

$$19. k_{fg}(x) = \left| \frac{x (fg)'(x)}{(fg)(x)} \right| = \left| \frac{x f'(x) g(x) + x g'(x) f(x)}{f(x) g(x)} \right| \leq k_f(x) + k_g(x).$$

21. From Theorem 1.8 and using $\left| \frac{x - x^*}{x} \right| \leq \frac{1}{2}10^{-k+1}$, we have

$$\left| \frac{f(x) - f(x^*)}{f(x)} \right| \leq \frac{1}{2}10^{-k} = \frac{1}{2}10^{-(k+1)+1}.$$

Solutions from Chapter 2

5. (a) We use induction. Clearly, $x^* \in [x_0, y_0]$, $f(x_0) < 0$, and $f(y_0) \geq 0$. Assume that $x^* \in [x_{k-1}, y_{k-1}]$ and $f(x_{k-1}) < 0$, $f(y_{k-1}) \geq 0$. If $f\left(\frac{x_{k-1} + y_{k-1}}{2}\right) < 0$, then $x_k = \frac{x_{k-1} + y_{k-1}}{2}$ and $y_k = y_{k-1}$. Thus, $f(x_k) < 0$ and $f(y_k) \geq 0$ and $x^* \in [x_k, y_k]$. If $f\left(\frac{x_{k-1} + y_{k-1}}{2}\right) > 0$, then $x_k = x_{k-1}$ and $y_k = \frac{x_{k-1} + y_{k-1}}{2}$. Thus $f(x_k) < 0$ and $f(y_k) \geq 0$ and $x^* \in [x_k, y_k]$. The proof thus follows by induction.

(b) Note that

$$y_k - x_k = \frac{1}{2}(y_{k-1} - x_{k-1}) = \frac{1}{2^k}(y_0 - x_0) = \frac{b - a}{2^k}.$$

However, $|x^* - x_k| \leq |x_k - y_k| \leq \frac{b-a}{2^k}$, since $x^* \in [x_k, y_k]$.

12. Let $g(x) = b + \epsilon h^2(x)$. Then $x_{k+1} = b + \epsilon h^2(x_k)$ is the same as the fixed point iteration $x_{k+1} = g(x_k)$. Clearly $g : \mathbb{R} \rightarrow \mathbb{R}$. Now consider, for $x, y \in \mathbb{R}$,

$$\begin{aligned} |g(x) - g(y)| &= |\epsilon| |h^2(x) - h^2(y)| = |\epsilon| |h(x) + h(y)| |h(x) - h(y)| \\ &\leq 2ML|\epsilon||x - y| < 1 \cdot |x - y|. \end{aligned}$$

Thus $g(x)$ is a contraction, so there exists a unique z such that $z = g(z)$ and $\{x_k\}_{n=0}^{\infty}$ converges to z for any $x_0 \in \mathbb{R}$, by the Contraction Mapping Theorem.

14. We have

$$\begin{aligned} g(x) &= \frac{1}{3} \left[\frac{x^3}{3} - x^2 - \frac{5}{4}x + 4 \right], \\ g'(x) &= \frac{1}{3} \left[x^2 - 2x - \frac{5}{4} \right], \quad g''(x) = \frac{1}{3}[2x - 2]. \end{aligned}$$

First, note that $g(x)$ has maximum and minimum values on $[0, 2]$ at $x = 0$, $x = 2$, or where $g'(x) = 0$. (However, $g'(x) = 0$ at $x = \frac{5}{2}$ and

$x = -\frac{1}{2}$.) Thus, $\frac{1}{18} = g(2) \leq g(x) \leq g(0) = \frac{4}{3}$ for any $x \in [0, 2]$. Thus, $g(x) \in [0, 2]$ if $x \in [0, 2]$ so $g : [0, 2] \rightarrow [0, 2]$. One may also note that

$$\max_{0 \leq x \leq 2} |g'(x)| = \max\{|g'(0)|, |g'(1)|, |g'(2)|\} = \frac{3}{4}.$$

Thus, g is a contraction on $[0, 2]$. The result now follows from the Contraction Mapping Theorem.

23. Since $f \in C^1$ and $x_k \rightarrow r$ we have $f(x_k) \rightarrow f(r)$ and $f'(x_k) \rightarrow f'(r)$. Hence, taking the limits, we get

$$\lim_{k \rightarrow \infty} x_k = \lim_{k \rightarrow \infty} x_{k-1} - \lim_{k \rightarrow \infty} \frac{f(x_{k-1})}{f'(x_{k-1})} \implies r = r - \frac{f(r)}{f'(r)} \implies f(r) = 0.$$

Also, using Taylor expansion, we have $f(x_k) = f(r) + f'(x)(x_k - r)$, where x is between x_k and r . Then, $f(r) = 0$ gives

$$|x_k - r| \leq \max_{x \in [a, b]} \frac{|f(x_k)|}{|f'(x)|}.$$

25. Consider the function $f(x) = R - 1/x$. Then, $f'(x) = 1/x^2$, and Newton's method is

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \implies x_{k+1} = x_k - \frac{R - 1/x}{1/x^2},$$

which can be simplified to yield $x_{k+1} = x_k(2 - Rx_k)$.

26. Let $f(x) = \int_0^x e^{t^2} dt - 1$. We have a single nonlinear equation to solve to obtain x^* such that $f(x^*) = 0$. There is a unique zero because $f(0) < 0$, $f(1) > 0$, and $f'(x) > 0$ for $x \in \mathbb{R}$. One way to numerically solve the nonlinear equation is to use Newton's method, which for this equation is

$$x_{k+1} = x_k - \frac{\int_0^{x_k} e^{t^2} dt - 1}{e^{x_k^2}}.$$

The integral can be computed numerically using some numerical integration method described in Chapter 6.

Solutions from Chapter 3

1. Let A be an invertible matrix. For $x \neq 0$, let $y = A^{-T}x \neq 0$. Since A is positive definite, we have $(y^T A y)^T > 0 \implies [(A^{-T}x)^T A (A^{-T}x)]^T > 0 \implies [x^T A^{-1} A A^{-T} x]^T = (x^T A^{-T} x)^T = x^T A^{-1} x > 0$. Hence, A^{-1} is also positive definite.

2. Let A be a symmetric positive definite $n \times n$ matrix. For a given index j , consider a nonzero vector $\vec{x} = (x_i)$ be defined by $x_j = 1$ and $x_i = 0$, if $i \neq j$. Then we have $\vec{x}^T A \vec{x} > 0 \implies a_{jj} > 0$ for each $j = 1, 2, \dots, n$.
7. (a) Let $F = -A^{-1}B$ then $A + B = A(I - F)$. Suppose $I - F$ is singular, then for $x \neq 0$, $(I - F)x = 0 \implies \|Fx\| = \|x\| \implies \|x\| \leq \|x\| \|F\| \implies \|F\| \geq 1$, i.e., $\|A^{-1}B\| \geq 1$ a contradiction. Hence $I - F = I + A^{-1}B$ is nonsingular and since A is also nonsingular $\implies A + B = A(I - F)$ is nonsingular. Moreover,

$$\begin{aligned} \|(A + B)^{-1}\| &= \|(A(I - F))^{-1}\| = \|(I - F)^{-1}A^{-1}\| \\ &\leq \|(I - F)^{-1}\| \|A^{-1}\| \leq \frac{\|A^{-1}\|}{1 - \|F\|} = \frac{\|A^{-1}\|}{1 - r} \end{aligned}$$

(b)

$$\begin{aligned} \|(A + B)^{-1} - A^{-1}\| &= \|(A + B)^{-1}(I - (A + B)A^{-1})\| \\ &= \|(A + B)^{-1}(I - AA^{-1} - BA^{-1})\| \\ &= \|(A + B)^{-1}(-BA^{-1})\| \\ &\leq \|(A + B)^{-1}\| \|B\| \|A^{-1}\| \\ &\leq \frac{\|B\| \|A^{-1}\|^2}{1 - r} \quad \text{using (a)} \end{aligned}$$

12. (a) $B = A - C = A(I - A^{-1}C)$. Also, $\|A^{-1}C\|_2 \leq \|A^{-1}\|_2 \|C\|_2 = 10^3 \cdot 10^{-4} = \frac{1}{10} < 1$. Hence, $\rho(A^{-1}C) < 1$, and $(I - A^{-1}C)$ is nonsingular. Hence B is nonsingular.

(b) $x = A^{-1}b$ and $z = B^{-1}b$. Therefore,

$$\begin{aligned} \|x - z\|_2 &= \|A^{-1}b - B^{-1}b\|_2 = \|A^{-1}b - (I - A^{-1}C)^{-1}A^{-1}b\|_2 \\ &\leq \|I - (I - A^{-1}C)^{-1}\|_2 \|A^{-1}b\|_2 \\ &\leq \|(I - A^{-1}C)^{-1}\|_2 \|A^{-1}C\|_2 \|x\|_2 \\ &\leq \frac{\|A^{-1}C\|_2}{1 - \|A^{-1}C\|_2} \|x\|_2. \end{aligned}$$

Using part (a), we find that $\|x - z\|_2 \leq \frac{1}{9} \|x\|_2$. Hence, $c = \frac{1}{9}$.

13. (a) Let $A = 5(I - H)$, where

$$H = \begin{cases} 0 & \text{if } i = j, \\ -\frac{1}{5} & \text{if } i = j + 1 \text{ or } i = j - 1, \\ 0 & \text{otherwise.} \end{cases}$$

Note that $\|H\|_\infty = \frac{2}{5}$, so $\rho(H) \leq \|H\|_\infty < 1$. Therefore, $(I - H)$ is nonsingular, so A is nonsingular.

(b) Since H satisfies

$$\frac{1}{1 + \|H\|_\infty} \leq \|(I - H)^{-1}\|_\infty \leq \frac{1}{1 - \|H\|_\infty},$$

part (a) gives

$$A^{-1} = \frac{1}{5}(I - H)^{-1} \implies \|A^{-1}\|_\infty = \frac{1}{5}\|(I - H)^{-1}\|_\infty.$$

Hence, $\frac{1}{7} \leq \|A^{-1}\|_\infty \leq \frac{1}{3}$.

14. (a) We have

$$\begin{aligned} \|I - AB_k\| &= \|I - AB_{k-1} - AB_{k-1}(I - AB_{k-1})\| \\ &= \|(I - AB_{k-1})^2\| \\ &\leq \|I - AB_{k-1}\|^2 \leq \|I - AB_{k-1}\|^4 \\ &\leq \dots \leq \|I - AB_{k-1}\|^{2^k} = c^{2^k}. \end{aligned}$$

(b) Since $\|I - AB_0\| = c < 1$, we have

$$\begin{aligned} \|(I - (I - AB_0))^{-1}\| &\leq \frac{1}{1 - \|I - AB_0\|} \\ \implies \|B_0^{-1}A^{-1}\| &\leq \frac{1}{1 - c}. \end{aligned}$$

Then

$$\|A^{-1}\| = \|B_0B_0^{-1}A^{-1}\| \leq \frac{\|B_0\|}{1 - c}.$$

Finally, using the part (a),

$$\|A^{-1} - B_k\| = \|A^{-1}(I - AB_k)\| \leq \|A^{-1}\| \|I - AB_k\|,$$

and the result follows.

15. Since λ is an eigenvalue of A , by definition there exists an eigenvector x such that $Ax = \lambda x$. Thus,

$$Ix - cAx = x - c\lambda x \implies (I - cA)x = (1 - c\lambda)x.$$

Therefore, $1 - c\lambda$ is an eigenvalue of $(I - cA)$ with corresponding eigenvector x .

16. Since $\rho(A) < 1$, then by problem 15, none of the eigenvalues of $I - A$ can equal zero. Thus, $(I - A)$ is nonsingular. By directly multiplying, we have $(I - A)(I + A + A^2 + \dots + A^N) = I - A^{N+1}$. Hence,

$$\sum_{i=0}^N A^i = (I + A + A^2 + \dots + A^N) = (I - A)^{-1}(I - A^{N+1})$$

because $I - A$ is nonsingular.

32. Consider

$$\begin{aligned} y^T A y &= \sum_{i=1}^n \sum_{j=1}^n y_i a_{ij} y_j = \sum_{i=1}^n \sum_{j=1}^n \int_0^1 y_i e^{ix} y_j e^{jx} dx \\ &= \int_0^1 \left(\sum_{i=1}^n y_i e^{ix} \right) \left(\sum_{j=1}^n y_j e^{jx} \right) dx. \end{aligned}$$

Letting $f(x) = \sum_{j=1}^n y_j e^{jx}$, we then have $\int_0^1 (f(x))^2 dx \geq 0$. Let $z = e^x$. Then $\sum_{j=1}^n y_j e^{jx} = \sum_{j=1}^n y_j z^j$ is a polynomial of degree n in z . But $\sum_{j=1}^n y_j z^j = 0$ only if $y_j = 0$ for $j = 1, \dots, n$. Thus $y^T A y > 0$ if $y \neq 0$. Hence, A is positive definite and therefore has a Cholesky factorization.

37. $A = I - xx^T \implies A^T = A$ and

$$A^T A = (I - xx^T)^T (I - xx^T) = I - 2xx^T + xx^T xx^T.$$

Note that $x^T x = \|x\|_2^2 = 2$. We then have $A^T A = I \implies A^{-1} = A^T$. The condition number then is given by

$$k_2(A) = \|A^{-1}\|_2 \|A\|_2 = \|A^T\|_2 \|A\|_2 = \|A\|_2^2 = \rho(A^T A) = \rho(I) = 1.$$

48. A is strictly diagonally dominant. Hence, the Gauss-Seidel and Jacobi methods converge.

49. Suppose $\|M^{-1}N\| < 1$ then $I - M^{-1}N$ is nonsingular. Since M is also nonsingular, $M(I - M^{-1}N)$ is nonsingular. This implies that

$$A = M - N = M(I - M^{-1}N)$$

is nonsingular, which is a contradiction. Hence, $\|M^{-1}N\| \geq 1$.

50. Since A is positive definite,

$$\det(A) > 0 \implies \gamma\alpha - \beta^2 > 0 \implies \frac{\beta}{\sqrt{\gamma\alpha}} < 1.$$

The Jacobi matrix corresponding to the given matrix A is

$$J = \begin{pmatrix} 0 & \frac{\beta}{\alpha} \\ \frac{\beta}{\gamma} & 0 \end{pmatrix}.$$

The eigenvalues of J are $\pm\beta/\sqrt{\gamma\alpha} < 1$. Hence, $\rho(J) < 1$, which implies that the Jacobi method converges.

52. Consider

$$T_1 = -(L + D)^{-1}U = -\begin{pmatrix} 2 & 0 \\ 1 & 2 \end{pmatrix}^{-1} \begin{pmatrix} 0 & -5 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & \frac{5}{2} \\ 0 & -\frac{5}{4} \end{pmatrix}.$$

Thus $\rho(T_1) = \frac{5}{4} > 1$, so the method is not convergent for $\sigma = 1$. Consider

$$T_{\frac{1}{2}} = \left(\frac{1}{2}L + D\right)^{-1} \left(\frac{1}{2}D - \frac{1}{2}U\right) = \begin{pmatrix} 2 & 0 \\ \frac{1}{2} & 2 \end{pmatrix}^{-1} \begin{pmatrix} 1 & \frac{5}{2} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{5}{4} \\ -\frac{1}{8} & \frac{3}{16} \end{pmatrix}.$$

Furthermore, $\rho(T_{1/2}) = 1/2$. Hence, the method is convergent for $\sigma = 1/2$.

Solutions from Chapter 4

3. Since $g^* \in \mathcal{P}^n$ is the least squares approximation to f , we have

$$g^*(x) = \sum_{i=0}^n (\varphi_i, f) \varphi_i(x).$$

Now let $p(x) = \sum_{j=0}^n p_j \varphi_j(x)$. Then

$$\begin{aligned} \int_{-1}^3 (f(x) - g^*(x))p(x)dx &= (f, p) - (g^*, p) \\ &= \sum_{j=0}^n p_j (f, \varphi_j) - \sum_{i=0}^n (\varphi_i, f) (p, \varphi_i) \\ &= \sum_{j=0}^n p_j (f, \varphi_j) - \sum_{i=0}^n (f, \varphi_i) p_i \\ &= 0. \end{aligned}$$

4. The polynomials are

$$\begin{aligned} q_0(x) &= \varphi_0, \\ q_1(x) &= \varphi_1 - (\varphi_1, \varphi_0) \frac{\varphi_0}{\|\varphi_0\|^2}, \\ q_2(x) &= \varphi_2 - (\varphi_2, \varphi_0) \frac{\varphi_0}{\|\varphi_0\|^2} - (\varphi_2, q_1) \frac{q_1}{\|q_1\|^2}. \end{aligned}$$

Using $\{\varphi_0, \varphi_1, \varphi_2\} = \{1, x, x^2\}$ and $(f, g) = \int_0^1 f(x)g(x)dx$, we get

$$q_0(x) = 1, \quad q_1(x) = x - \frac{1}{2}, \quad q_2(x) = x^2 - x + \frac{1}{6}.$$

The least squares approximation of $f(x) = x^{\frac{1}{2}}$ on $[0, 1]$ then is

$$\sum_{i=0}^2 c_i q_i, \quad \text{where } c_i = \frac{(f, q_i)}{(q_i, q_i)}.$$

Using this, we get $c_0 = \frac{2}{3}$, $c_1 = \frac{4}{5}$, $c_2 = -\frac{4}{7}$. Hence,

$$f(x) = \frac{2}{3}q_0(x) + \frac{4}{5}q_1(x) - \frac{4}{7}q_2(x) = \frac{1}{35}(6 + 48x - 20x^2).$$

6. (a) Follows from Theorems 4.5 and 4.6.

(b) One can easily find that $P_1(x) = \frac{3}{2} - \frac{x}{2}$. Moreover, $f''(\xi) = \frac{2}{\xi^3}$. Thus,

$$f(x) - P_1(x) = \frac{1}{x} - \frac{3}{2} + \frac{x}{2} = \frac{1}{2}(x-1)(x-2)\frac{2}{\xi(x)^3}.$$

Thus, solving this equation for ξ we get $\xi(x) = (2x)^{1/3}$. Since this is an increasing function we find that

$$1.2599 \approx 2^{1/3} \leq \xi(x) \leq 4^{1/3} \approx 1.5874, \quad 1 \leq x \leq 2$$

8. Let $A = (a_{ij}) = (w_i, w_j)$. Note that $a_{ij} = a_{ji}$. For $x \neq 0$, we have

$$\begin{aligned} x^T A x &= \sum_{j=1}^n x_j \left(\sum_{i=1}^n a_{ij} x_i \right) = \sum_{j=1}^n x_j \left(\sum_{i=1}^n (w_i, w_j) x_i \right) \\ &= \left(\sum_{i=1}^n w_i x_i, \sum_{j=1}^n w_j x_j \right) \\ &= z^T z > 0, \end{aligned}$$

where $z = \sum_{i=1}^n w_i x_i$.

11. Given $\epsilon > 0$, the Weierstrass approximation theorem states that there exists a polynomial p such that $\|f - p\|_\infty < \epsilon/2$. Let N be the degree of this polynomial p . Now assume $n \geq N$ and consider

$$\begin{aligned} |E_n(f)| &= |E_n(f) - E_n(p)| \\ &= \left| \int_0^1 (f(x) - p(x)) dx - \sum_{i=1}^n w_i (f(x_i) - p(x_i)) \right| \\ &\leq \int_0^1 |f(x) - p(x)| dx + \sum_{i=1}^n |f(x_i) - p(x_i)| w_i \\ &\leq 2\|f - p\|_\infty \leq \epsilon \end{aligned}$$

when $n \geq N$.

14. Using the transformation $x = 2(t + 1)$, we get

$$\begin{aligned} P_3(t) &= 8t^3 - 12t^2 - 88t - 63 \\ &= -63T_0 - 88T_1 - 6(T_2 + T_0) - 2(T_3 + 3T_1) \\ &= -69T_0 - 82T_1 - 6T_2 + 2T_3, \end{aligned}$$

where $-1 \leq t \leq 1$ and $T_i(x)$ is the corresponding Chebyshev polynomial. If we truncate the polynomial at T_2 , we have

$$\max_{-1 \leq t \leq 1} |P_3(t) - (-69T_0 - 82T_1 - 6T_2)| = \max_{-1 \leq t \leq 1} |2T_3| = 2.$$

The approximation is

$$P_2(t) = -69T_0 - 82T_1 - 6T_2 = -63 - 82t - 12t^2,$$

which has maximum absolute error $\delta = 2$. Substituting $t = (x - 2)/2$, we get $P_2(x) = -3x^2 - 29x + 7$.

15. (a) Note that $l_i(x_l) = \delta_{il}$ and $\hat{l}_j(y_m) = \delta_{jm}$. Thus

$$p(x_l, y_m) = \sum_{i=1}^n \sum_{j=1}^n c_{ij} l_i(x_l) \hat{l}_j(y_m) = c_{lm},$$

Therefore, $p(x, y)$ interpolates $f(x, y)$ at the n^2 points if $c_{lm} = f(x_l, y_m)$ for $m, l = 1, 2, \dots, n$.

- (b) Note that $\sum_{i=1}^n l_i(k)$ interpolates 1 at $x_1, x_2, \dots, x_n$. Define the $(n - 1)$ -degree polynomial $p(x)$ by $p(x) = \sum_{i=1}^n l_i(x) - 1$. Since $p(x_i) = 0$ for $i = 1, 2, \dots, n$, we have $p(x) = 0$. Thus $\sum_{i=1}^n l_i(x) = 1$. But

$$\sum_{i=1}^n \sum_{j=1}^n l_i(x) \hat{l}_j(y) = \sum_{i=1}^n l_i(x) \sum_{j=1}^n \hat{l}_j(y) = 1.$$

26. The piecewise polynomials $s_1(x)$ and $s_2(x)$ must satisfy $s_2(0) = s_1(0) = 1 + c$, $s_2'(0) = s_1'(0) = 3c$, $s_2''(0) = s_1''(0) = 6c$. From these, we obtain $s_2(x) = 1 + c + 3cx + 3cx^2 + Ax^3$. Since $s(x)$ is a natural cubic spline, we also have $s_2''(1) = 0 \implies A = -c$. Moreover, we also require $s(1) = -1 \implies s_2(1) = -1$. This then yields $1 + c + 3c + 3c - c = -1 \implies c = -\frac{1}{3}$.

27. We have

$$\begin{aligned} \max_{0 \leq x \leq 1} |\psi(x) - f(x)| &= \max_{0 \leq i \leq N-1} \left\{ \max_{x_i \leq x \leq x_{i+1}} |f(x_i) - f(x)| \right\} \\ &\leq \max_{0 \leq i \leq N-1} \left\{ \max_{x_i \leq x \leq x_{i+1}} L|x_i - x| \right\} \\ &\leq Lh. \end{aligned}$$

35. Starting with $e^x = \sum_{j=0}^{\infty} \frac{x^j}{j!}$, consider

$$\left(\sum_{j=0}^{\infty} \frac{x^j}{j!} \right) (1 + b_1x + b_2x^2) - (a_0 + a_1x) = \sum_{i=0}^{\infty} d_i x^i.$$

Setting $d_i = 0$ for $i = 0, 1, 2, 3$, we get $1 - a_0 = 0$, $1 + b_1 - a_1 = 0$, $\frac{1}{2} + b_1 + b_2 = 0$, $\frac{1}{6} + \frac{b_1}{2} + b_2 = 0$. Solving these equations we get $a_0 = 1$, $a_1 = \frac{1}{3}$, $b_1 = -\frac{2}{3}$, $b_2 = \frac{1}{6}$, and

$$e^x \approx \frac{1 + \frac{x}{3}}{1 - \frac{2x}{3} + \frac{1}{6}x^2}.$$

Solutions from Chapter 5

3. Using Gerschgorin's Circle Theorem, we find that $|\lambda| \leq \max\{\frac{9}{20}, \frac{7}{12}, \frac{8}{15}\}$. Hence, $\rho(A) < 1$.
7. Let $q^{(0)} = \sum_{i=1}^n c_i x_i$, where $\{x_i\}_{i=1}^n$ are the orthonormal eigenvectors of A , where x_1 through x_{m_1} $m_1 \geq 1$ correspond to $\lambda_1 = 1$, and where x_{m_1+1} to $x_{m_1+m_2}$, $m_2 \geq 1$ correspond to $\lambda_2 = -1$. Then,

$$\begin{aligned} q^{(r)} &= \left(\sum_{i=1}^n \lambda_i^r x_i \right) \\ &= \sum_{i=1}^{m_1} c_i x_i + (-1)^r \left(\sum_{i=m_1+1}^{m_1+m_2} c_i x_i \right) + \sum_{i=m_1+m_2+1}^n c_i \lambda_i^r x_i, \end{aligned}$$

where the last sum is interpreted to be zero if $m_1 + m_2 = n$. Then, given $\epsilon > 0$, there is an N such that

$$\left\| \sum_{i=m_1+m_2+1}^n c_i \lambda_i^r x_i \right\| < \frac{\epsilon}{2}$$

for $r > N$, since $|\lambda_j| < 1$ for $j \geq m_1 + m_2$. Thus

$$q^{(r+1)} - q^{(r)} = 2(-1)^{r+1} \left(\sum_{i=m_1+1}^{m_1+m_2} c_i x_i \right) + \sum_{i=m_1+m_2+1}^n c_i (\lambda_i^{r+1} - \lambda_i^r) x_i,$$

which then gives

$$\|q^{(r+1)} - q^{(r)}\| \geq 2 \sqrt{\sum_{i=m_1+1}^{m_1+m_2} c_i^2} - \epsilon$$

for $r > N$, where ϵ can be made arbitrarily small. Hence, the method fails to converge. Indeed, for r large,

$$q^{(r)} \approx \left(\sum_{i=1}^{m_1} c_i x_i \right) + (-1)^r \left(\sum_{i=m_1+1}^{m_1+m_2} c_i x_i \right),$$

and $q^{(r)}$ oscillates and never converges.

8. Notice that the method given in this problem is a form of deflation technique. Let $z^{(0)} = \sum_{i=1}^n c_i x_i$. Then,

$$q^{(0)} = z^{(0)} - \left(z^{(0)T} x_1 \right) x_1 = \sum_{i=1}^n c_i x_i - \left(\sum_{i=1}^n c_i x_i^T x_1 \right) x_1 = \sum_{i=2}^n c_i x_i.$$

In addition,

$$\begin{aligned} q^{(r+1)} &= Aq^{(r)} - \left((Aq^{(r)})^T x_1 \right) x_1 \\ &= \sum_{i=2}^n c_i \lambda_i^{r+1} x_i = \lambda_2^{r+1} \left[c_2 x_2 + \sum_{i=3}^n c_i \left(\frac{\lambda_i}{\lambda_2} \right)^{r+1} x_i \right]. \end{aligned}$$

Thus, $\lim_{r \rightarrow \infty} \left(\frac{q^{(r+1)}}{\lambda_2^{r+1}} \right) = c_2 x_2$, and $q^{(r)}$ goes in the direction x_2 as r increases.

Solutions from Chapter 6

5. The Taylor polynomial of degree 2 for f , with error term, gives

$$f(x_0 + h) = f(x_0) + f'(x_0)h + f''(x_0)\frac{h^2}{2} + f'''(\xi_0)\frac{h^3}{6}.$$

Similarly,

$$f(x_0 + 2h) = f(x_0) + f'(x_0)(2h) + f''(x_0)2h^2 + f'''(\xi_1)\frac{8h^3}{6}.$$

Substituting these expansions into

$$\left| f'(x_0) - \frac{1}{h} \left[-\frac{3}{2}f(x_0) + c_1 f(x_0 + h) + c_2 f(x_0 + 2h) \right] \right|,$$

and setting coefficients equal to zero except for the h^2 coefficient yields

$$c_1 + c_2 = \frac{3}{2}, \quad c_1 + 2c_2 = 1, \quad \frac{c_1}{2} + 2c_2 = 0,$$

which can be solved to give $c_1 = 2$, $c_2 = -\frac{1}{2}$.

6. (a) Taylor series expansions give

$$f(x_0 + h) = f(x_0) + f'(x_0)h + f''(x_0)\frac{h^2}{2} + f'''(x_0)\frac{h^3}{3!} + \dots$$

and

$$f(x_0 - h) = f(x_0) - f'(x_0)h + f''(x_0)\frac{h^2}{2} - f'''(x_0)\frac{h^3}{3!} + \dots$$

Then,

$$\frac{f(x_0 + h) - f(x_0 - h)}{2h} = f'(x_0) + \frac{f'''(x_0)}{3!}h^2 + \frac{f^{(5)}(x_0)}{5!}h^4 + \dots$$

Thus,

$$c_k = \frac{f^{(2k+1)}(x_0)}{(2k+1)!} \quad \text{for } k = 1, 2, \dots$$

- (b) Let $\alpha_1 = -\frac{1}{3}$ and $\alpha_2 = \frac{4}{3}$. (Get this by setting the lower-order coefficients in the error expansion to zero.)
18. (a) Letting $f(x) = 1$ and $f(x) = x$ yields $a_0 = \frac{5}{36}$ and $a_1 = \frac{1}{9}$.
- (b) Using the transformation $t = xh$, we get

$$\int_0^h g(t) \, t \ln(h/t) \, dt = \int_0^1 g(xh) \, (xh) \, \ln(1/x) \, h dx.$$

From part (a), this becomes

$$h^2 \left(\frac{5}{36}g(0) + \frac{1}{9}g(h) \right).$$

19. Let $I = \int_a^b f(x) \, dx$. Then

$$\begin{aligned} I(h) &= I + c_1 h + c_2 h^2 + O(h^3) \\ 2I\left(\frac{h}{2}\right) &= 2I + c_1 h + c_2 \frac{h^2}{2} + O(h^3) \\ 3I\left(\frac{h}{3}\right) &= 3I + c_1 h + c_2 \frac{h^2}{3} + O(h^3) \end{aligned}$$

Manipulating these equations one can obtain

$$\frac{9}{2}I\left(\frac{h}{2}\right) - 4I\left(\frac{h}{3}\right) + \frac{1}{2}I(h) = I + O(h^3).$$

21. The formula will be exact for polynomials of degree less than equal to 5. Hence, choosing $f(x) = 1, x, x^2, x^3, x^4, x^5$ yields the following equations:

$$\begin{aligned} \sum_{i=1}^3 A_i &= \pi, & \sum_{i=1}^3 A_i x_i &= 0, \\ \sum_{i=1}^3 A_i x_i^2 &= \frac{\pi}{2}, & \sum_{i=1}^3 A_i x_i^3 &= 0, \\ \sum_{i=1}^3 A_i x_i^4 &= \frac{3\pi}{8}, & \sum_{i=1}^3 A_i x_i^5 &= 0. \end{aligned}$$

Solving these equations gives

$$\int_{-1}^1 \frac{1}{\sqrt{1-x^2}} f(x) dx \approx \frac{\pi}{3} \left[f\left(\frac{-\sqrt{3}}{2}\right) + f(0) + f\left(\frac{\sqrt{3}}{2}\right) \right].$$

The x_i 's may be computed either with the technique in Remark 6.11 on page 347 or by observing that the Chebyshev polynomials are orthogonal with respect to the dot product

$$(f, g) = \int_{-1}^1 \frac{1}{\sqrt{1-x^2}} f(x) g(x) dx.$$

In particular,

$$T_3(x) = 4x^3 - 3x = 0 \implies x = \frac{-\sqrt{3}}{2}, 0, \frac{\sqrt{3}}{2}.$$

$$\begin{aligned} 23. \quad & \int_0^1 \int_0^1 f(x, y) \, dx \, dy - h^2 \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} f(ih, jh) \\ &= \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \int_{ih}^{(i+1)h} \int_{jh}^{(j+1)h} (f(x, y) - f(ih, jh)) \, dx \, dy \\ &= \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \int_{ih}^{(i+1)h} \int_{jh}^{(j+1)h} \\ & \quad \cdot \left(\frac{\partial f(\mu_i, \xi_j)}{\partial x} (x - ih) + \frac{\partial f(\mu_i, \xi_j)}{\partial y} (y - jh) \right) \, dx \, dy. \end{aligned}$$

Using the Mean Value Theorem for integrals, this becomes:

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \left(\frac{\partial f(\hat{\mu}_i, \hat{\xi}_j)}{\partial x} \frac{h^3}{2} + \frac{\partial f(\tilde{\mu}_i, \tilde{\xi}_j)}{\partial y} \frac{h^3}{2} \right).$$

We then use the Intermediate Value Theorem to obtain

$$\frac{h}{2} \left[\frac{\partial f(\check{\mu}_i, \check{\xi}_j)}{\partial x} + \frac{\partial f(\check{\mu}_i, \check{\xi}_j)}{\partial y} \right],$$

which gives the result.

25. The integral $\int_{-1}^1 \int_{-1}^1 x^i y^j \, dx \, dy = 0$ when i and/or j is odd. For $i = j = 0$, the method is exact. For $f(x, y) = x^2 y^2$, we obtain $\frac{4}{9} = 4\alpha^4 \implies \alpha = \pm \frac{1}{\sqrt[4]{3}}$.

26. For Monte Carlo calculations,

$$\text{prob} \left(\left| \frac{1}{n} \sum_{i=1}^n f(x_i) - \int_0^1 f(x) dx \right| \leq \frac{\lambda \sigma}{\sqrt{n}} \right) \leq \frac{1}{\sqrt{2\pi}} \int_{-\lambda}^{\lambda} e^{-\frac{x^2}{2}} dx.$$

We require

$$\frac{1}{\sqrt{2\pi}} \int_{-\lambda}^{\lambda} e^{-\frac{x^2}{2}} dx \geq 0.997.$$

Solving this equation by using the error function (such as using the MATLAB function `erf` or by looking it up in a table), we see that $\lambda = 3$ is sufficient. We thus obtain the inequality $\frac{3\sigma}{\sqrt{n}} \leq 0.001$ for this problem. But

$$\sigma^2 = \int_0^1 e^{2x} dx - \left(\int_0^1 e^x dx \right)^2 \in [0.24203, 0.24204].$$

Hence, $\sqrt{n} \geq 3000\sigma$ or $n \geq 9 \times 10^6 \sigma^2$, and $9 \times 10^6 \sigma^2 \leq 5.28 \times 10^5$. Thus, for this problem, to reduce the error to less than 0.001 with probability 0.997, $n \geq 2.19 \times 10^6$ is sufficient.

Solutions from Chapter 7

3. Since $f(x, y)$ does not satisfy a Lipschitz condition for $x \geq 0$, there is no guarantee that Euler's method will converge to the solution. Indeed, for this problem, there are an infinite number of solutions. One solution is $y(x) = 0$ which Euler's method appears to approximate. However

$$y(x) = \begin{cases} 0, & 0 \leq x \leq a, \\ \left(\frac{2}{3}(x-a)\right)^{\frac{3}{2}}, & x \geq a \end{cases}$$

is a solution for any positive number a .

4. Clearly $f(t, y) = \frac{t}{9} \cos(2y) + t^2$ is continuous in t and y . We need to show that $f(t, y)$ satisfies a Lipschitz condition in y . We have

$$|f(t, y) - f(t, \tilde{y})| = \left| \frac{t}{9} (\cos(2y) - \cos(2\tilde{y})) \right| \leq \frac{10}{9} |2 \sin(2\xi)| |y - \tilde{y}|$$

using the Mean Value Theorem. Thus,

$$|f(t, y) - f(t, \tilde{y})| \leq \frac{20}{9} |y - \tilde{y}|.$$

Therefore, there is a unique solution to this initial-value problem for $0 \leq t \leq 10$.

6. Notice that

$$y(t_{n+1}) = y(t_n) + hf(y(t_n), t_n) + \frac{h^2}{2} y''(\xi_n).$$

Let $E_n = |y(t_n) - y_n|$. Then $E_{n+1} \leq E_n + hLE_n + \delta + \frac{h^2}{2}M$. Using a telescoping argument, we get

$$\begin{aligned} E_{n+1} &\leq \left(\delta + \frac{h^2}{2}M \right) [1 + (1 + hL) + (1 + hL)^2 + \dots + (1 + hL)^n] \\ &\leq \left(\delta + \frac{h^2}{2}M \right) \frac{(1 + hL)^{n+1} - 1}{hL}. \end{aligned}$$

This then implies

$$E_N \leq \left(\delta + \frac{h^2}{2}M \right) \frac{(1 + hL)^N - 1}{hL} \leq \frac{\delta}{h} \left[\frac{e^L - 1}{L} \right] + h \left[\frac{M}{2L} (e^L - 1) \right].$$

10. Let $z_1 = y$, $z_2 = y'$, $z_3 = y''$. Then we have $z'_1 = z_2$, $z'_2 = z_3$, $z'_3 = t + 2tz_3 + 2t^2z_1$ with $z_1(1) = 1$, $z_2(1) = 2$, $z_3(1) = 3$. Euler's method has the form

$$z_{i+1} = z_i + hf(t_i, z_i), \quad i = 0, 1, 2, \dots$$

We have $h = 0.1$, $t_0 = 1$, $z_0 = [1 \ 2 \ 3]^T$. Thus,

$$\begin{aligned} z_1 &= z_0 + 0.1f(1, z_0) = [1.2 \ 2.3 \ 3.9]^T \quad \text{and} \\ z_2 &= z_1 + 0.1f(1.1, z_1) = [1.43 \ 2.69 \ 5.1584]^T. \end{aligned}$$

19. Consider

$$\begin{aligned}
 |y_{i+1} - \hat{y}_{i+1}| &= \left| y_i + \frac{h}{2}[f(t_{i+1}, y_{i+1}) + f(t_i, y_i)] - \hat{y}_i \right. \\
 &\quad \left. - \frac{h}{2}[f(t_{i+1}, \hat{y}_{i+1}) + f(t_i, \hat{y}_i)] \right| \\
 &\leq |y_i - \hat{y}_i| + \frac{h}{2}|f(t_{i+1}, y_{i+1}) - f(t_{i+1}, \hat{y}_{i+1})| \\
 &\quad + \frac{h}{2}|f(t_i, y_i) - f(t_i, \hat{y}_i)| \\
 &\leq |y_i - \hat{y}_i| + \frac{h\lambda}{2}|y_{i+1} - \hat{y}_{i+1}| + \frac{h\lambda}{2}|y_i - \hat{y}_i| \\
 &\leq \frac{1 + h\lambda/2}{1 - h\lambda/2}|y_i - \hat{y}_i| \\
 &\leq e^{3\lambda h/2}|y_i - \hat{y}_i|
 \end{aligned}$$

for $\lambda h \leq 1$. Continuing,

$$e^{3\lambda h/2}|y_i - \hat{y}_i| \leq (e^{3\lambda h/2})^{i+1}|y_0 - \hat{y}_0| = (e^{3\lambda t_{i+1}/2})|y_0 - \hat{y}_0|.$$

Hence,

$$|y_i - \hat{y}_i| \leq e^{3\lambda t_i/2}|y_0 - \hat{y}_0| \leq e^K|y_0 - \hat{y}_0|, \quad \text{where } K = 3\lambda T/2.$$

22. (a) Note that the exact solution satisfies

$$y(t_{k+2}) + \alpha_1 y(t_{k+1}) + \alpha_0 y(t_k) = h\beta y'(t_{k+2}) + h\tau$$

where τ is the local truncation error. Hence,

$$h\tau = y(t_{k+2}) + \alpha_1 y(t_{k+1}) + \alpha_0 y(t_k) - h\beta y'(t_{k+2}).$$

Using the Taylor expansions of $y(t_{k+2})$, $y(t_{k+1})$, and $y'(t_{k+2})$ about $t = t_k$, we have

$$\begin{aligned}
 h\tau &= [1 + \alpha_1 + \alpha_0]y(t_k) + h[2 + \alpha_1 - \beta]y'(t_k) \\
 &\quad + h^2[2 + \frac{\alpha_1}{2} - 2\beta]y''(t_k) + O(h^3).
 \end{aligned}$$

For the method to be second order, we need

$$1 + \alpha_1 + \alpha_0 = 0, \quad 2 + \alpha_1 - \beta = 0, \quad \text{and} \quad 2 + \frac{\alpha_1}{2} - 2\beta = 0,$$

which yields $\beta = \frac{2}{3}$, $\alpha_1 = -\frac{4}{3}$, $\alpha_0 = \frac{1}{3}$.

(b) For consistency note that $\alpha_0 + \alpha_1 + \alpha_2 = 0$ where $\alpha_2 = 1$ and also $\alpha_1 + 2\alpha_2 = \beta$. Hence the method is consistent.

(c) For stability consider $\rho(z) = \frac{1}{3} - \frac{4}{3}z + z^2 = 0$ which implies $z = \frac{1}{3}, 1$. Hence $|z_i| \leq 1$ and the method is stable.

Solutions from Chapter 8

3. There is a $\hat{\delta}$ such that

$$\begin{aligned}\|F(y) - F(x)\| &= \|F(y) - F(x) - F'(x)(y - x) + F'(x)(y - x)\| \\ &\leq \|F(y) - F(x) - F'(x)(y - x)\| + \|F'(x)\| \|y - x\| \\ &\leq \|y - x\| + \|F'(x)\| \|y - x\|\end{aligned}$$

for all $\|y - x\| \leq \hat{\delta}$. (Existence of a $\hat{\delta}$ such that

$$\|F(y) - F(x) - F'(x)(y - x)\| < \|y - x\|$$

for $\|y - x\| < \hat{\delta}$ follows from the definition of the Fréchet derivative.) Thus

$$\|F(y) - F(x)\| \leq (1 + \|F'(x)\|) \|y - x\| < \epsilon$$

when

$$\|y - x\| \leq \min \left\{ \hat{\delta}, \frac{\epsilon}{1 + \|F'(x)\|} \right\}$$

Thus, $\|F(y) - F(x)\| \leq \epsilon$ when $\|x - y\| < \delta$. Hence, F is continuous at x .

7. Consider $x^{(k+1)} = G(x^{(k)}) = Bx^{(k)} + b$. Then there exists a matrix norm $\|\cdot\|$ such that $\|B\| \leq \rho(B) + \epsilon = \gamma < 1$. $G: \mathbb{R}^n \rightarrow \mathbb{R}^n$ and G is a contraction on $\mathbb{R}^n$, since $\|G(x) - G(y)\| = \|B(x - y)\| \leq \|B\| \|x - y\|$ for $x, y \in \mathbb{R}^n$. Therefore, by the Contraction Mapping Theorem, G has a unique fixed point $x^* \in \mathbb{R}^n$ and $\{x^{(k)}\}_{k=0}^\infty$ converges to x^* . Now suppose that $\rho(B) \geq 1$. Let λ be an eigenvalue of magnitude $\rho(B)$, i.e., $|\lambda| \geq 1$, and let x be its associated eigenvector. Then $Bx = \lambda x$. Suppose that $x^{(0)} = x$ and $b = 0$. Then $x^{(k)} = \lambda^k x^{(0)}$ for $k = 0, 1, 2, \dots$. This implies that $\rho(B) < 1$ is necessary for convergence for any $x^{(0)} \in \mathbb{R}^n$.

14. We have

$$F(x) = \nabla f(x) = \begin{bmatrix} e^{x_1} - x_2 + 2x_1 - 1 \\ e^{x_2} - x_1 + 2x_2 - 1 \end{bmatrix} \quad \text{and}$$

$$\nabla^2 f(x) = \begin{bmatrix} e^{x_1} + 2 & -1 \\ -1 & e^{x_2} + 2 \end{bmatrix} = F'(x).$$

Note that:

- (i) $F(x)$ is continuously differentiable on $\mathbb{R}^2$, since all elements of $F(x)$ are continuous.

- (ii) $F'(x)$ exists on $\mathbb{R}^2$ and is nonsingular on $\mathbb{R}^2$ because $\nabla^2 f(x)$ is strictly diagonally dominant on $\mathbb{R}^2$.
- (iii) One may verify that $(F'(x))^{-1} > 0$ on $\mathbb{R}^2$ (for example, by computing the inverse symbolically and examining the resulting elements).
- (iv) $\nabla f(x) = 0$ has a solution at $x_1 = x_2 = 0$.
- (v) Note that

$$F(y) - F(x) = \begin{bmatrix} e^{y_1} - e^{x_1} - (y_2 - x_2) + 2(x_1 - y_1) \\ e^{y_2} - e^{x_2} - (y_1 - x_1) + 2(x_2 - y_2) \end{bmatrix},$$

and also

$$F'(x)(y - x) = \begin{bmatrix} (e^{x_1} + 2)(y_1 - x_1) - (y_2 - x_2) \\ (e^{x_2} + 2)(y_2 - x_2) - (y_1 - x_1) \end{bmatrix}.$$

But $e^b - e^a \geq e^a(b - a)$ for all $a, b \in \mathbb{R}$, so F is convex. Therefore, the Newton Iterates converge for any $x^{(0)} \in \mathbb{R}^2$.

Solutions from Chapter 9

3. We have

$$\begin{aligned} B_{k+1}s &= \left(I - \frac{y s^T}{y^T s}\right) B_k \left(I - \frac{s y^T}{y^T s}\right) s + \frac{y y^T s}{y^T s} \\ &= \left(I - \frac{y s^T}{y^T s}\right) B_k \left(s - s \frac{y^T s}{y^T s}\right) + y \\ &= y. \end{aligned}$$

Thus, the Davidon–Fletcher–Powell method satisfies the quasi-Newton equation.

8. Note that

$$\begin{aligned} f(x_{k+1}) &= f(x_k - t_k \nabla f(x_k)) \\ &= \varphi_k(t_k) = \min_{t \in \mathbb{R}} \varphi_k(t) = \min_{t \in \mathbb{R}} f(x_k - t \nabla f(x_k)) \\ &\leq f(x_k). \end{aligned}$$

If $\nabla f(x_k) = 0$, then $x_{k+1} = x_k$, so $f(x_{k+1}) = f(x_k)$. If $\nabla f(x_k) \neq 0$, this implies that $t = 0$ is a minimum of $\varphi_k(t)$. Therefore, $\varphi'_k(0) = 0$. But $\varphi'_k(0) = (\nabla f(x_k))^T \nabla f(x_k)$ using the chain rule. Hence, $(\nabla f(x_k))^T \nabla f(x_k) = 0$, so $\nabla f(x_k) = 0$.

Solutions from Chapter 10

1. (a) Using $z = \frac{dy}{dx}$, we have the following system of first order equations:

$$\begin{aligned}\frac{dy}{dx} &= z = f(x, y, z), & y(2) &= 1 \\ \frac{dz}{dx} &= \frac{6y}{x^2}, = g(x, y, z) & z(2) &= 0\end{aligned}$$

Using Euler's method, we get

$$y_{i+1} = y_i + hf(x_i, y_i, z_i), \quad z_{i+1} = z_i + hg(x_i, y_i, z_i).$$

With the given initial conditions, one can then calculate $y_1 = 1$, $z_1 = \frac{3}{2}$, $y_2 = \frac{5}{2} = y(4)$.

- (b) Starting from $z(2) = 3$ and repeating the steps in part (a), we get $y_1 = 4$, $z_1 = \frac{9}{2}$, $y_2 = \frac{17}{2} = y(4)$.
- (c) Let the guess be G to obtain the desired result $y(4) = 8$. Using Lagrange interpolation, we get

$$y(4) = \frac{G-3}{0-3} \left(\frac{5}{2} \right) + \frac{G-0}{3-0} \left(\frac{17}{2} \right).$$

Solving for the guess, we get $G = \frac{11}{4}$.

3. First note that

$$\begin{aligned}\frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1}))}{h^2} - p(x_i) \frac{y(x_{i+1}) - y(x_i)}{h} - q(x_i)y(x_i) \\ = r(x_i) + \tau_i,\end{aligned}$$

where

$$\tau_i = \frac{h^2}{12} y^{(4)}(\xi_i) - p(x_i) \frac{h}{2} y^{(2)}(\eta_i)$$

for some $\xi_i, \eta_i \in [x_{i-1}, x_{i+1}]$. Subtracting the difference equation from this expression, letting $\epsilon_i = y(x_i) - u_i$, and rearranging the expression yields

$$\begin{aligned}(3 + q(x_i)h^2)\epsilon_i &= (1 - hp(x_i))\epsilon_{i+1} + (1 + hp(x_i))\epsilon_i + \epsilon_{i-1} \\ &\quad - \frac{h^4}{12} y^{(4)}(\xi_i) + \frac{h^3}{2} p(x_i) y^{(2)}(\eta_i).\end{aligned}$$

Assuming $h < 1/P^*$ where

$$P^* = \max_{a \leq x \leq b} |p(x)| \quad \text{and} \quad Q_* = \min_{a \leq x \leq b} q(x) > 0,$$

letting $|\epsilon| = \max_i |\epsilon_i|$, and simplifying the resulting expression yields the result.

10. We have

$$f(t_i) = g(t_i) + \int_0^1 K(t_i, s)f(s)ds$$

and

$$F_i = g(t_i) + h \sum_{j=0}^N w_j K(t_i, t_j) F_j.$$

Subtracting, we get

$$\begin{aligned} f(t_i) - F_i &= \int_0^1 K(t_i, s)f(s)ds - h \sum_{j=0}^N w_j K(t_i, t_j)f(t_j) \\ &\quad + h \sum_{j=0}^N w_j K(t_i, t_j)(f(t_j) - F_j), \end{aligned}$$

which implies

$$\begin{aligned} |f(t_i) - F_i| &\leq ch^p + h \sum_{j=0}^N w_j |K(t_i, t_j)| |f(t_j) - F_j| \\ &\leq ch^p + \frac{1}{2} \sum_{j=0}^N h w_j |f(t_j) - F_j| \\ &\leq ch^p + \frac{1}{2} \max_j |f(t_j) - F_j|. \end{aligned}$$

The result then follows by taking the maximum over all i and combining the absolute value terms.

References

- [1] Edward J. Allen. *Modeling With Itô Stochastic Differential Equations*. Springer, Dordrecht, Netherlands, 2007.
- [2] Eugene L. Allgower and Kurt Georg. *Numerical Continuation Methods: An Introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1990.
- [3] Bradley K. Alpert. Wavelets and other bases for fast numerical linear algebra. In C. K. Chui, editor, *Wavelets: A Tutorial in Theory and Applications*, pages 181–216. Academic Press, San Diego, CA, USA, 1992.
- [4] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK's User's Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.
- [5] Krzysztof Apt. *Principles of Constraint Programming*. Cambridge University Press, New York, NY, USA, 2003.
- [6] N. S. (Asai) Asaithambi. *Numerical Analysis Theory and Practice*. Harcourt Brace College Publishers, Orlando, Florida, February 1995.
- [7] Kendall E. Atkinson. *A Survey of Numerical Methods for Fredholm Integral Equations of Second Kind*. SIAM, Philadelphia, PA, USA, 1976.
- [8] Kendall E. Atkinson. *An Introduction to Numerical Analysis*. Wiley, New York, NY, USA, second edition, 1989.
- [9] Owe Axelsson and Vincent A. Barker. *Finite Element Solution of Boundary Value Problems, Theory and Computation*. Academic Press, Orlando, Florida, 1984. Republished by SIAM in 2001, ISBN: 0-89871-499-0.
- [10] Christopher T. H. Baker and Graham S. Hodgson. Asymptotic expansions for integration formulas in one or more dimensions. *SIAM J. Numer. Anal.*, 8(2):473–480, June 1971.
- [11] Richard J. Bauer. *Genetic Algorithms and Investment Strategies*. Wiley, New York, NY, USA, 1994.
- [12] Arthur T. Benjamin. Sensible rules for remembering duals—the S-O-B method. *SIAM Rev.*, 37(1):85–87, 1995.

- [13] Martin Berz, Kyoko Makino, Khodr Shamseddine, and Weishi Wan. *Modern Map Methods in Particle Beam Physics*. Academic Press, San Diego, 1999.
- [14] Garrett Birkhoff and Carl de Boor. Error bounds for spline interpolation. *Journal of Mathematics and Mechanics*, 13:827–836, 1964.
- [15] William L. Briggs and Van Emden Henson. Wavelets and multigrid. *SIAM Journal on Scientific Computing*, 14(2):506–510, March 1993.
- [16] Eliot W. Cheney. A survey of methods for rational approximation, with particular reference to a new method based on a formula of Darboux. *SIAM Review*, 5(3):219–231, July 1963.
- [17] Elliot W. Cheney. *Introduction to Approximation Theory*. McGraw–Hill, New York, NY, USA, 1966.
- [18] Earl A. Coddington. *An Introduction to Ordinary Differential Equations*. Prentice–Hall Mathematics Series. Prentice–Hall, Englewood Cliffs, NJ, USA, 1961.
- [19] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301, 1965.
- [20] Constantin Corduneanu. *Principles of Differential and Integral Equations*. Chelsea Publishing, New York, NY, USA, 1977.
- [21] George F. Corliss and Louis B. Rall. Adaptive, self-validating numerical quadrature. *SIAM Journal on Scientific and Statistical Computing*, 8(5):831–847, September 1987.
- [22] Philip J. Davis and Philip Rabinowitz. *Numerical Integration*. Blaisdell Publishing, Waltham, MA, USA, 1967. A second edition, from 1984, is available.
- [23] James W. Demmel. *Applied Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1997.
- [24] John E. Dennis, Jr. and Robert B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, volume 16 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 1996.
- [25] John E. Dennis, Jr. and Robert B. Schnabel. Least change secant updates for quasi-Newton methods. *SIAM Review*, 21(4):443–459, October 1979.
- [26] Kaisheng Du and Ralph Baker Kearfott. The cluster problem in global optimization: The univariate case. *Computing (Suppl.)*, 9:117–127, 1992.

- [27] Iain S. Duff, Albert M. Erisman, C. William Gear, and John K. Reid. Sparsity structure and Gaussian elimination. *SIGNUM Newsl.*, 23(2):2–8, 1988.
- [28] Bruno Dupire. *Monte Carlo Methodologies and Applications for Pricing and Risk Management*. Risk Books, London, 1998.
- [29] Michael C. Ferris, Olvi L. Mangasarian, and Stephen J. Wright. *Linear Programming with MATLAB (MPS-SIAM Series on Optimization)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.
- [30] George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler. *Computer Methods for Mathematical Computations*. Prentice–Hall Professional Technical Reference, 1977.
- [31] Curtis F. Gerald and Patrick O. Wheatley. *Applied Numerical Analysis*. Addison–Wesley, Reading, MA, USA, fifth edition, 1994.
- [32] P. E. Gill, W. Murray, and M. Wright. *Practical Optimization*. Academic Press, New York, NY, USA, 1981.
- [33] Gene H. Golub and James M. Ortega. *Scientific Computing and Differential Equations: An Introduction to Numerical Methods*. Academic Press, Orlando, FL, USA, 1991.
- [34] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. Johns Hopkins University Press, third edition, 1996.
- [35] David Gottlieb and Steven A. Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*, volume 26 of *Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, PA, USA, 1977.
- [36] Andreas Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Number 19 in *Frontiers in Appl. Math.* SIAM, Philadelphia, PA, 2000.
- [37] Max D. Gunzburger. Numerical analysis: Lecture notes, 1979. University of Tennessee, Knoxville.
- [38] Randy L. Haupt and Sue Ellen Haupt. *Practical genetic algorithms*. Wiley, New York, NY, USA, 1998.
- [39] Eugene Isaacson and Herbert Bishop Keller. *Analysis of Numerical Methods*. Dover, New York, NY, USA, 1994.
- [40] Alan Jennings. *Matrix Computation for Engineers and Scientists*. Wiley, New York, NY, USA, 1977.
- [41] Bush Jones. A note on the T_{+m} transformation. *Nonlinear Analysis: Theory, Methods and Applications*, 6(4):303–305, April 1982.

- [42] N. Karmarkar. A new polynomial-time algorithm for linear programming. In *STOC '84: Proceedings of the sixteenth annual ACM symposium on theory of computing*, pages 302–311, New York, NY, USA, 1984. ACM.
- [43] Ralph Baker Kearfott. Abstract generalized bisection and a cost bound. *Mathematics of Computation*, 49(179):187–202, July 1987.
- [44] Ralph Baker Kearfott. *Rigorous Global Search: Continuous Problems*. Number 13 in Nonconvex optimization and its applications. Kluwer Academic Publishers, Dordrecht, Netherlands, 1996.
- [45] Ralph Baker Kearfott. On proving existence of feasible points in equality constrained optimization problems. *Math. Program.*, 83(1):89–100, 1998.
- [46] Ralph Baker Kearfott and Kaisheng Du. The cluster problem in multivariate global optimization. *Journal of Global Optimization*, 5:253–265, 1994.
- [47] Ralph Baker Kearfott and Siriporn Hongthong. Validated linear relaxations and preprocessing: Some experiments. *SIAM J. on Optimization*, 16(2):418–433, 2005.
- [48] Ralph Baker Kearfott and G. William Walster. On stopping criteria in verified nonlinear systems or optimization algorithms. *ACM Transactions on Mathematical Software*, 26(3):373–389, September 2000.
- [49] David Kincaid and Ward Cheney. *Numerical Analysis: Mathematics of Scientific Computing*. Brooks / Cole, Pacific Grove, California, third edition, 2002.
- [50] Jack D. Lambert. *Computational Methods for Ordinary Differential Equations*. Wiley, New York, NY, USA, 1973.
- [51] Jack D. Lambert. The initial value problem for ordinary differential equations. In David A. H. Jacobs, editor, *The State of the Art in Numerical Analysis*, pages 451–500. Academic Press, New York, NY, USA, 1977.
- [52] Peter Lancaster and Kestutis Salkauskas. *Curve and Surface Fitting: An Introduction*. Academic Press, London, 1986.
- [53] Peter Linz. *Analytical and Numerical Methods for Volterra Equations*. Studies in Applied Mathematics. SIAM, Philadelphia, 1985.
- [54] David G. Luenberger. *Investment Science*. Oxford University Press, New York, NY, USA, 1998.
- [55] Neil Madras. *Lecture Notes on Monte Carlo Methods*. Fields Institute Monographs. American Mathematical Society, Providence, Rhode Island, 2002.

- [56] Gurii I. Marchuk and Vladimir V. Shaidurov. *Difference Methods and Their Extrapolations*. Springer, New York, NY, USA, 1983.
- [57] R. E. Moore, R. B. Kearfott, and M. J. Cloud. *Introduction to Interval Analysis*. SIAM, Philadelphia, PA, 2009.
- [58] Ramon E. Moore. *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, PA, USA, 1979.
- [59] Alexander Morgan and Andrew Sommese. A homotopy for solving general polynomial systems that respects m-homogenous structures. *Appl. Math. Comput.*, 24(2):101–113, 1987.
- [60] Alexander P. Morgan. *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*. Prentice Hall, Englewood Cliffs, NJ USA, 1987.
- [61] John A. Nelder and Roger Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.
- [62] Arnold Neumaier. *Interval Methods for Systems of Equations*, volume 37 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, UK, 1990.
- [63] Arnold Neumaier and Oleg Shcherbina. Safe bounds in linear and mixed-integer programming. *Math. Prog.*, 99(2):283–296, March 2004.
- [64] Arnold Neumaier and Zuhe Shen. The Krawczyk operator and Kantorovich’s theorem. *Mathematical Analysis and Applications*, 149(2):437–443, July 1990.
- [65] T. A. J. Nicholson. *Optimization in Industry Volume 1: Optimization Techniques*. Longman Group, New York, NY, USA, 1971.
- [66] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, volume 63 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, PA, USA, 1992.
- [67] Ben Noble. The numerical treatment of nonlinear integral equations and related topics. In Philip M. Anselone, editor, *Nonlinear Integral Equations*. University of Wisconsin Press, Madison, WI, USA, 1964.
- [68] James. M. Ortega. *Numerical Analysis: A Second Course*. Academic Press, New York, NY, USA, 1972.
- [69] James M. Ortega. *Matrix Theory: A Second Course*. University Series in Mathematics. Plenum Presas, New York, NY, USA, 1987.
- [70] James M. Ortega and Werner C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.
- [71] John Derwent Pryce and George F. Corliss. Interval arithmetic with containment sets. *Computing*, 78(3):251–276, 2006.

- [72] Louis B. Rall. A comparison of the existence theorems of Kantorovich and Moore. *SIAM J. Numer. Anal.*, 17(1):148–161, February 1980.
- [73] Werner C. Rheinboldt. *Methods for Solving Systems of Nonlinear Equations*. SIAM, Philadelphia, PA, USA, 1974.
- [74] John R. Rice. *The Approximation of Functions*, volume 1. Addison–Wesley, Reading, MA, USA, 1964.
- [75] Theodore J. Rivlin. *An Introduction to the Approximation of Functions*. Blaisdell Publishing, Waltham, MA, USA, 1969.
- [76] Siegfried M. Rump. Verification methods for dense and sparse systems of equations. In Jürgen Herzberger, editor, *Topics in Validated Computations: proceedings of IMACS-GAMM International Workshop on Validated Computation, Oldenburg, Germany, 30 August–3 September 1993*, volume 5 of *Studies in Computational Mathematics*, pages 63–136, Amsterdam, The Netherlands, 1994. Elsevier.
- [77] Siegfried M. Rump. INTLAB–INTerval LABoratory. In Tibor Csendes, editor, *Developments in Reliable Computing: Papers presented at the International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics, SCAN-98, in Szeged, Hungary*, volume 5(3) of *Reliable Computing*, pages 77–104, Dordrecht, Netherlands, 1999. Kluwer Academic Publishers. URL: <http://www.ti3.tu-harburg.de/rump/intlab/>.
- [78] Youcef Saad. *Iterative Methods for Sparse Linear Systems*. PWS Kent Publishing Company, Boston, MA USA, 1996.
- [79] Hermann Schichl and Arnold Neumaier. Exclusion regions for systems of equations. *SIAM Journal on Numerical Analysis*, 42(1):383–408, 2004.
- [80] Martin H. Schultz. *Spline Analysis*. Prentice–Hall, Englewood Cliffs, NJ USA, 1973.
- [81] Christoph Schwab. *P- and Hp- Finite Element Methods. Theory and Applications to Solid and Fluid Mechanics*. Oxford University Press, New York, NY, USA, 1998.
- [82] Padmanabhan Seshaiyer and Manil Suri. hp submeshing via non-conforming finite element methods. *Computer Methods in Applied Mechanics and Engineering*, 189:1011–1030, September 2000.
- [83] Granville Sewell. *The Numerical Solution of Ordinary and Partial Differential Equations*. Wiley, New York, NY, USA, second edition, 2005.
- [84] Lawrence F. Shampine and Marilyn K. Gordon. *Computer Solution of Ordinary Differential Equations*. W. H. Freeman, San Fransisco, CA, USA, 1975.

- [85] Gilbert W. Stewart. *Introduction to Matrix Computations*. Academic Press, New York, NY, USA, 1973.
- [86] Josef Stoer and Roland Bulirsch. *Introduction to Numerical Analysis*. Springer, New York, 1980. A third edition, 2002, is available.
- [87] Gilbert Strang and George J. Fix. A Fourier analysis of the finite element variational method. In *Constructive aspects of Functional Analysis*, Rome, 1973. Edizione Cremonese.
- [88] Friedrich Stummel. Forward error analysis of Gaussian elimination. I. error and residual estimates. *Numerische Mathematik*, 46(3):365–395, June 1985.
- [89] Friedrich Stummel. Forward error analysis of Gaussian elimination. II. stability theorems. *Numerische Mathematik*, 46(3):397–415, June 1985.
- [90] Friedrich Stummel and Karl Hainer. *Introduction to Numerical Analysis*. Springer, New York, NY, USA, 1980. Translated by E. R. Dawson.
- [91] Leonard W. Swanson. *Linear Programming: Basic Theory and Applications*. McGraw–Hill, New York, NY, USA, 1980.
- [92] Barna Szabó and Ivo Babuska. *Finite Element Analysis*. Wiley, New York, NY, USA, 1991.
- [93] Mohit Tawarmalani and Nikolaos V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Kluwer Academic Publishers, Dordrecht, Netherlands, 2002.
- [94] John A. Tierney. *Differential Equations*. Allyn and Bacon, Boston, MA, USA, second edition, 1985.
- [95] Peter J. M. van Laarhoven and Emile H. L. Aarts. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, Norwell, MA, USA, 1987.
- [96] Richard S. Varga. *Matrix Iterative Analysis*. Springer, New York, NY, USA, second edition, 2000.
- [97] David S. Watkins. *Fundamentals of Matrix Computations*. Wiley, New York, NY, USA, 1991. A second edition is available, 2002.
- [98] Layne T. Watson, Stephen C. Billups, and Alexander P. Morgan. Algorithm 652: Hompack: a suite of codes for globally convergent homotopy algorithms. *ACM Trans. Math. Softw.*, 13(3):281–310, 1987.
- [99] Burton Wendroff. *Theoretical Numerical Analysis*. Academic Press, Englewood Cliffs, NJ, USA, 1966.
- [100] James Hardy Wilkinson. *Rounding Errors in Algebraic Processes*. Prentice–Hall, Englewood Cliffs, NJ, USA, 1963.

- [101] James Hardy Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1988.
- [102] Stephen J. Wright. *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1997.
- [103] David M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, New York, NY, USA, 1971.
- [104] Eberhard Zeidler. *Nonlinear Functional Analysis and its Applications*. Springer, New York, NY, USA, 1985.

Index

- A(0) stability, 425
- A-stable
 - method for an IVP, 423
- absolute error, 14
- absolute stability
 - methods for a scalar equation, 400
 - methods for systems, 421
 - predictor-corrector method, 418
- accuracy, order of
 - multistep method, 407
- Adams methods, 406, 417
- Adams–Bashforth methods, 406
- Adams–Moulton methods, 406
- adaptive quadrature, 374
- Aitken acceleration, 65
- approximation
 - best, 192
 - least squares, 198
 - minimax, 230
- Arnoldi’s Algorithm, 171
- asymptotic rate of convergence, 161
- automatic differentiation, 328
 - forward mode, 328
 - reverse mode, 330
- automatically verified, 519

- B-spline, 243
 - basis, 244
- back substitution, 106, 115
- backsolving, 106
- backward error analysis, 126
- Banach space, 197
- basic feasible point
 - of a linear program, 506
- basis
 - B-spline, 244
 - collocating, 212
 - Haar, 274
 - hat function, 239
 - Lagrange, 212
 - linear programming, 506
 - of a vector space, 89
- basis functions, 140
- Bernoulli number, 358
- Bernoulli numbers, 355
- Bernstein polynomials, 206
- best approximation, 192
- BFGS update, 496
- BFP, 535
- bifurcation point, 481
- big $\mathcal{O}$ notation, 5
- bisection
 - method of, 36
- black box function, 58
- black box system, 459, 479
- block SOR method, 161
- Bolzano–Weierstrass Theorem, 194
- bound constraints, 487
- boundary value problem, 535
- boundary value problems
 - finite difference methods, 540
 - shooting method, 536
- bounded variation, 250
- bounding step, 523
- branch and bound algorithm, 74, 374, 523
 - clustering effect, 530
 - for nonlinear systems of equations, 481
- branching step, 523
- Brouwer fixed point theorem, 471
- Broyden update, 474
- Broyden’s method, 475

- Broyden–Fletcher–Goldfarb–Shanno update, 496
- Cauchy polygon method, 385
- Cauchy sequence, 40, 197
- Cauchy–Schwarz inequality, 90
- central difference formula, 324
- Central Limit Theorem, 369
- centroid
 - of a simplex, 498
- chapeau functions, 547
- characteristic equation, 87
- characteristic polynomial, 291
- Chebyshev norm, 191
- Chebyshev polynomials, 217, 224
- Chebyshev’s Equi-Oscillation Theorem, 234
- Cholesky factorization, 117
- chop, 11
- clamped spline interpolant, 246
- clustering effect, branch and bound algorithm, 530
- code list, 329
- collocating basis, 212, 240
- companion matrix, 74, 170
- compatible matrix and vector norms, 96
- complete search algorithms, 519
- complete space, 40
- complex Fourier series, 250
- complex reflector, 135
- composite integration, 334
- condition
 - ill, 122
 - number
 - generalized, 180
 - of a function, 16
 - of a matrix, 123
 - perfect, 124
 - Wilkinson polynomial, 73
- consistency
 - of a k -step method, 409
 - of a method for solving an IVP, 391
- consistently ordered matrix, 158
- constrained optimization, 487
- constraint programming, 529
- constraint propagation, 527
- constraint satisfaction problem, 488
- continuation method, 481
- continuity
 - modulus of, 230
- contraction, 40
- Contraction Mapping Theorem, 442
 - in one variable, 40
- convergence
 - global, 454
 - iterative method for linear systems, 142
 - linear, 7
 - local, 454
 - of a sequence of matrices, 102
 - of a sequence of vectors, 92
 - of the SOR method, 147
 - order of, 7
 - quadratic, 7
 - rate
 - asymptotic, 161
 - semilocal, 454
 - superlinear, 8
- convex
 - function, 455
 - program, 488, 515
 - set, 444
 - strictly, 475
- correct rounding, 23
- critical point
 - of a constrained optimization problem, 502
 - of an unconstrained optimization problem, 492
- cubic spline, 243
- cyclic matrix, 158
- Daubechies scaling function, 278
- Davidon–Fletcher–Powell update, 496
- defective matrix, 292
- deflation
 - eigenvalues of matrices, 305
 - roots of polynomials, 72

- dependency, interval, 27
- derivative tensor, 441
- Descartes' rule of signs, 70
- DFP update, 496
- diagonally dominant, 150
 - irreducibly, 150
 - strictly, 111, 150
- differentiation
 - automatic, 328
- dilates, 270
- dimension
 - of a vector space, 89
- Dirichlet problem, 535
- discretization error, 387
- distance
 - in a normed space, 92
- divided difference
 - k -th order, 213
 - first order, 63, 213
 - Newton's backward formula, 215
 - Newton's formula, 214
 - second order, 63
- domain
 - frequency, 258
 - time, 258
- Doolittle algorithm, 185
- double QR algorithm, 312
- dual problem, 502
- dual variables, 502
- dynamic programming, 515
- eigenvalue, 86, 291
 - simple, 297
- eigenvector, 86, 291
- elementary row operations
 - for linear systems, 88
 - in the simplex method, 508
- equi-oscillation property
 - minimax, 234
- equilibration, row, 125
- equivalent norms, 93
- error
 - absolute, 14
 - backward analysis, 126
 - bound for a single-step method
 - for IVP, 393
 - forward analysis, 126
 - local truncation
 - for a single-step method for IVP's, 391
 - method, 9
 - relative, 14
 - roundoff, 9
 - roundout, 25
 - truncation, 9
- Euclidean norm, 90
- Euler's method, 385
- Euler–Maclaurin formula, 355
- Euler-Trapezoidal Method, 417
- excess width, 27
- Exchange Method of Remez, 235
- explicit method
 - for solving IVP's, 406
- explicit single-step method, 389
- extended precision, 389
- extended real numbers, 24
- Fast Fourier Transform, 256
- fathomed, 526
- feasible point, 507
- feasible set, 488
- feasible solution, 507
- FFT, 256
- Fibonacci sequence, 62
- filtering, 258
- finite difference methods, 540
- finite element method, 547
- fixed point, 39, 442
 - iteration method, 39
- floating point numbers, 10
- forward difference formula, 215, 323
- forward error analysis, 126
- forward mode, automatic differenti-
ation, 328
- Fourier series, 204
 - complex, 250
- Fréchet derivative, 440
- Fredholm integral equation
 - of the second kind, 554, 559

- existence and uniqueness, 560
 - numerical solution of, 562
- frequency domain, 258
- Fritz John conditions, 503
- Frobenius norm, 96
- full pivoting
 - Gaussian elimination, 114
- full rank matrix, 86
- Fundamental theorem of algebra, 69
- fundamental theorem of interval arithmetic, 26
- GA's, 520
- Galerkin method, 545, 549
 - for Fredholm integral equations of the second kind, 563
 - functional analysis setting, 549
- Gauss–Legendre quadrature, 348
- Gauss–Seidel method, 143
- Gauss–Seidel–Newton method, 461
 - different from Newton–Gauss–Seidel Method, 462
- Gaussian elimination, 105
 - full pivoting, 114
 - partial pivoting, 114
- Gaussian quadrature, 343
 - 2-point, 344
 - definition, 344
 - error term, 349
 - Gauss–Legendre rules, 348
- generalized condition number, 180
- Generalized Rolle's Theorem, 216
- genetic algorithms, 520
- Gerschgorin's Circle Theorem, 293
 - for Hermitian matrices, 296
- Givens rotation, 137, 316
- global convergence, 454
- global minimizer, 489
- global optimization, 489, 518
- global truncation error, 387
- golden mean, 491
- golden section search, 490
- Gram matrix, 198
- Gram–Schmidt process, 138, 201
 - modified, 139, 171
- graph of a matrix, 157
- Haar basis, 274
- Halton sequence, 371
- harmonic analysis, 258
- hat functions, 239, 547
- Hermite interpolating polynomial, 220
- Hermite interpolation, 220
- Hermite polynomials, 224
- Hermitian matrix, 87, 295
- Hessenberg matrix, 307
- Hessian matrix, 492
- heuristic, 497
- Hilbert matrix, 125
- Hilbert space, 197
- homotopy method, 480
 - predictor-corrector method, 481
- Horner's method, 71
- Householder transformation, 133, 305
- HUGE, 20
- identity matrix, 86
- IEEE arithmetic, 19
- ill-conditioned, 122
- implicit method
 - for solving IVP's, 406
- implicit Simpson's method, 405
- implicit trapezoid method, 434
- improper integrals, 365
- improved Euler method, 397
- independent, linearly, 89
- infeasible linear program, 513
- infimum, 194
- infinite integrals, 365
- initial value problem, 381
- inner product, 91
 - real space, 195
- integration, 333
 - composite, 334
 - infinite, 365
 - midpoint rule, 341
 - Monte Carlo, 369
 - multiple, 364
 - singular, 365
- interior point methods, 507

- interpolating polynomial
 - Hermite, 220
 - Lagrange form, 210, 212, 338
 - Newton form, 214
- interpolation
 - hermite, 220
- interval arithmetic
 - fundamental theorem of, 26
 - operational definitions, 24
- interval dependency, 27
- interval extension
 - first order, 28
 - mean value, 33
 - multivariate mean value, 467
 - second order, 28
- interval Gauss–Seidel method
 - nonlinear, 465
- interval Newton
 - operator, 464
 - univariate, 54
- interval Newton method
 - multivariate, 463
 - quadratic convergence of, 57
 - univariate, 55
- inverse
 - of a matrix, 86
- inverse midpoint matrix, 154
- inverse power method, 303
- invertible matrix, 86
- irreducible, 150
- irreducible graph, 157
- irreducibly diagonally dominant, 150
- iterative refinement, 127
- IVP, 381
- Jacobi diagonalization, 315
- Jacobi method, 143
 - for computing eigenvalues, 315
 - nonlinear, 461
- Jacobi rotation, 316
- Jacobi–Newton Method, 461
- Jacobian matrix, 440
- Kantorovich Theorem, 454
- Kantorovich theorem, 51
- Karmarkar’s algorithm, 507
- kernel
 - of an integral equation, 554
- Krawczyk
 - operator, 471
- Krawczyk method
 - multivariate, 470
 - univariate, 81
- Kronecker delta function, 92
- Krylov subspace, 169
- Kuhn–Tucker equations, 501
- Kuhn–Tucker point, 502
- L-2 norm, 192
- Lagrange
 - basis, 210, 212
 - polynomial interpolation, 210, 212, 324, 338
- Laguerre polynomials, 224
- Lanczos Algorithm, 172
- Lax–Milgram lemma, 550
- least squares
 - approximation, 140, 198, 279
 - general least squares problem, 222
 - norm, 192
- left singular vector, 175
- Legendre polynomials, 203, 223
- Leibnitz rule, 264
- Lemaréchal’s technique, 532
- line search, 490
 - golden section, 490
- linear convergence, 7
- linear least squares problem, 279
- linear program, 487, 488
 - infeasible, 513
 - standard form, 504
 - unbounded, 513
- linear programming
 - interior point methods, 507
- linear relaxation, 527
- linearly independent, 89
- Lipschitz condition, 40, 382, 555
- Lipschitz matrix, 463
- local convergence, 454
- local minimizer, 489

- local truncation error
 - finite difference methods for bound-ary value problems, 542
 - of a method for solving IVP's, 391
- low discrepancy sequence, 371
- LU
 - decomposition, 109
 - factorization, 109
- machine constants, 20
- machine epsilon, 20
- mag, 156
- magnitude (of an interval), 156
- mantissa, 10
- matrix norm, 95
 - compatible, 96
 - Frobenius, 96
 - induced, 97
 - natural, 97
- max norm, 191
- mean value interval extension, 33
 - multivariate, 467
- mean value theorem
 - for integrals, 2
 - multivariate, 441
 - univariate, 3
- method error, 9
- method of bisection, 36
- midpoint method
 - for solution of initial value problems, 390
- midpoint rule
 - for quadrature, 341, 342, 349
- mig, 156
- mignitude (of an interval), 156
- minimax approximation, 230
- minimax equi-oscillation property, 234
- Miranda's Theorem, 468
- mixed boundary conditions, 544
- modified Euler method, 397
- modified Gram–Schmidt procedure, 139
- modified Gram–Schmidt process, 171
- modulus of continuity, 230
- monic polynomial, 218
- Monte Carlo integration, 369
- Monte Carlo method
 - quasi, 371
- Moore–Penrose pseudo-inverse, 176
- Muller's method, 63
- multi-stage decision processes, 515
- multiple integrals, 364
- multistep method, 405
- multivariate interval Newton operator, 464
- multivariate mean value theorem, 441
- NaN, 20
- natural or induced matrix norm, 97
- natural spline, 246
- near minimax approximation, 235
- Nelder–Mead simplex method, 497
- Newton's backward difference formula, 215
- Newton's divided difference formula, 214
- Newton's forward difference formula, 215
- Newton's method
 - convergence of, 50
 - multivariate, 447
 - local convergence of, 450
 - univariate, 49
- Newton–Cotes formulas, 336
 - closed, 336
 - open, 336
- Newton–Gauss–Seidel Method, 460
 - different from Method, 462
- Newton–Gauss–Seidel method
 - different from Gauss–Seidel–Newton Method, 462
- Newton–Kantorovich Theorem, 454
- Newton–SOR iteration, 1-step, 460
- node
 - of a graph, 157
- nondefective matrix, 292
- nonlinear interval Gauss–Seidel method, 465
- nonlinear Jacobi method, 461

- nonlinear program, 487
- nonsingular matrix, 86
- nonsmooth optimization, 532
- norm
 - L^2 , 192
 - Chebyshev, 191
 - equivalent, 93
 - Euclidean, 90
 - important ones on $\mathbb{C}^n$, 90
 - least squares, 192
 - matrix, 95
 - compatible, 96
 - Frobenius, 96
 - induced, 97
 - natural, 97
 - max, 191
 - of a vector space, 88
 - uniform, 191
- normal distribution
 - standard, 214
- normal equations, 140, 198
- normed vector space, 88
- not a number, 20
- NP-complete problem, 488
- objective function, 487
- operator overloading, 333
- optimization
 - constrained, 487
 - convex, 488
 - unconstrained, 487
- optimizer, 488
- optimizing point, 488
- optimum, 488
- order
 - of a single-step method for solving an IVP, 391
 - of convergence, 7
- order of accuracy
 - multistep method, 407
 - predictor-corrector method, 417
- origin shift, 308
- orthogonal complement, 199
- orthogonal decomposition, 132
- orthogonal polynomials, 222, 345
- orthogonal projection, 200
- Ostrowski, a lemma of, 449
- outward rounding, 25
- overflow, 20
- overloading, operator, 333
- overrelaxation factor, 145
- Padé approximant, 260
- Padé methods
 - for IVP's, 424
- partial pivoting
 - Gaussian elimination, 114
- pattern search algorithms, 497
- perfectly conditioned, 124
- perpendicular (from a point to a set), 200
- Perron–Frobenius theorem, 150
- plane rotation, 137, 316
- Poincaré's inequality, 553
- polynomial time algorithm, 488
- positive
 - definite, 87
 - semi-definite, 87
- preconditioning, 154
- predictor-corrector method
 - for a homotopy method, 481
 - for systems of ordinary differential equations, 416
- primal variables, 502
- product formula, 364
- projection operator, 199
- property A, 158
- pseudo-inverse, 176
- QR
 - decomposition, 132
 - factorization, 132
 - method, 307
 - convergence of, 309
 - double, 312
 - with origin shifts, 308
- quadratic convergence, 7
- quadratic programming, 514
- quadrature, 333
 - composite, 334

- Gaussian, 343
 - 2-point, 344
 - definition, 344
 - error term, 349
 - Gauss–Legendre rules, 348
 - midpoint rule, 342
 - midpoint rule, 341, 349
 - Newton–Cotes, 336
 - product formula, 364
 - rectangle rule, 364
 - Simpson’s rule, 342
 - symmetric rule, 363
 - trapezoidal rule, 342
- quasi-Monte Carlo method, 371
- quasi-Newton
 - Davidon–Fletcher–Powell update, 496
 - equation, 473
 - method, 473
- quasi-random sequence, 371
- R-stage Runge–Kutta method, 397
- rank
 - of a matrix, 86
- rational approximation, 259
- Rayleigh quotient, 300
- real inner product space, 195
- rectangle rule, 364
- reducible, 150
- reflector
 - complex, 135
- relative error, 14
- relaxation, 524
 - linear, 527
- relaxation direction, 164
- Remez Algorithm, 235
- residual vector, 127, 163
- Richardson extrapolation, 354
- right singular vector, 175
- Rolle’s Theorem, 216
 - Generalized, 216
- Romberg integration, 355
- round, 11
 - down, 19
 - to nearest, 19
 - to zero, 19
 - up, 19
- rounding modes, 19
- roundoff error, 9
 - in Gaussian elimination, 126
- roundout error, 25
- row equilibration, 125
- Runge’s function, 219, 284
- Runge–Kutta method, 396
 - fourth order classic, 398
 - R-stage, 397
 - stability of, 398
- scaling function
 - Daubechies, 278
- Schur decomposition, 101, 292
- Schwarz inequality, 90
- search tree, 526
- secant equation, 473
- secant method, 59
 - convergence of, 60
- secant update, 479
- semi-definite, 87
- semilocal convergence, 454
- sequence
 - Cauchy, 197
- Sherman–Morrison formula, 476
- shooting method, 536
- significant digits, 17
- similarity transformation, 292
- simple eigenvalue, 297
- simplex method
 - of linear programming, 507
 - of Nelder and Mead, 497
- Simpson’s method
 - for IVP, 405
- Simpson’s rule, 342
- simultaneous iteration, 318
- single use expression, 27
- single-step methods, 389
- singular integrals, 365
- singular vector
 - left, 175
 - right, 175
- slope matrix, 464

- smoothing polynomials, 281
- smoothness, 373
- solution set, 131
- SOR
 - matrix, 146
 - method, 145
- SOR method
 - block, 161
 - convergence of, 147
- span, 89
- sparse matrix, 157
- spectral radius, 87, 291
- spectrum, 291
- spline
 - B-, 243
 - clamped, 246
 - cubic, 243
 - natural, 246
- stability
 - A(0), 425
 - predictor-corrector method, 418
 - Runge–Kutta methods, 398
- standard form, linear program, 504
- standard normal distribution, 214
- steepest descent method, 493
- Steffensen's method, 68
- Stein's Theorem, 147
- Stein–Rosenberg theorem, 149
- stiff
 - system of ODE's, 419, 422
- strictly convex, 475
- strongly connected directed graph, 157
- subdistributivity, 25
- subspace
 - Krylov, 169
 - of a vector space, 89
- subspace iteration, 318
- successive overrelaxation, 145
- successive relaxation method, 143
- SUE, 27
- superlinear convergence, 8
- symmetric matrix, 87
- symmetric quadrature rule, 363
- synthetic division, 71
- tape, 329
- Taylor polynomial
 - approximation by, 209
 - multivariate, 442
- Taylor series methods
 - for solving IVP's, 395
- Taylor's theorem, 2
- tensor
 - derivative, 441
- time domain, 258
- TINY, 20
- total step method, 143
- tractable problems, 488
- trapezoid method
 - implicit, 434
- trapezoidal rule, 340, 342
- triangle inequality, 89, 90
 - for 2-norm, 91
 - when strict inequality, 485
- triangular
 - decomposition, 109
 - factorization, 109
- truncation error, 9
 - global, 387
- Tschebycheff polynomials, 217
- two-cyclic matrix, 158
- two-point compactification, 24
- unbounded linear program, 513
- unconstrained optimization, 487
- underflow, 20
- underrelaxation factor, 145
- uniform norm, 191
- unitary matrix, 124
- update
 - quasi-Newton, 495
- Vandermonde matrix, 211, 338
- variation, bounded, 250
- vector space
 - normed, 88
- Volterra integral equation
 - of the second kind, 554
 - numerical solution of, 557
- wavelet, 273

weak formulation, 544

Weierstrass approximation theorem,
205

Wilkinson polynomial, 73, 82

Wilkinson Prize, 82

Mathematics

Classical and Modern Numerical Analysis: Theory, Methods and Practice

provides a sound foundation in numerical analysis for more specialized topics, such as finite element theory, advanced numerical linear algebra, and optimization. It prepares graduate students for taking doctoral examinations in numerical analysis.

The text covers the main areas of introductory numerical analysis, including the solution of nonlinear equations, numerical linear algebra, ordinary differential equations, approximation theory, numerical integration, and boundary value problems. Focusing on interval computing in numerical analysis, it explains interval arithmetic, interval computation, and interval algorithms. The authors illustrate the concepts with many examples as well as analytical and computational exercises at the end of each chapter.

This advanced, graduate-level introduction to the theory and methods of numerical analysis supplies the necessary background in numerical methods so that readers can apply the techniques and understand the mathematical literature in this area. Although the book is independent of a specific computer program, MATLAB® code will be available on the CRC Press website to illustrate various concepts.

Features

- Provides a clear and solid introduction to the theory and application of computational methods for applied mathematics problems
- Helps prepare readers for doctoral examinations in numerical analysis
- Presents the most important advanced aspects of numerical linear algebra, finite element theory, approximation theory, optimization, and integral equations
- Covers interval computation methods in numerical analysis
- Includes fully worked out solutions for selected problems
- Offers the MATLAB files on www.crcpress.com



CRC Press

Taylor & Francis Group
an informa business

www.crcpress.com

6000 Broken Sound Parkway, NW
Suite 300, Boca Raton, FL 33487
270 Madison Avenue
New York, NY 10016
2 Park Square, Milton Park
Abingdon, Oxon OX14 4RN, UK

C9157

ISBN: 978-1-4200-9157-1



9 781420 091571